APPLICATION OF BAYESIAN NETWORKS TO MIDCOURSE MULTI-TARGET TRACKING

MICHAEL KOVACICH

PRESENTED AUGUST 3, 1989 TO THE PANEL ON ADVANCED CONCEPTS.
LMSC, SUNNYVALE, CA. BOHDAN BALKO, CHAIRMAN

K9-7246/060

Accession Number: 5496

Publication Date: Aug 03, 1989

Title: Application of Bayesian Networks to Midcourse Multi-Target Tracking

Personal Author: Kovacich, M.

Comments on Document: Presented August 3, 1989, to the Panel on Advanced Concepts. LMSC, Sunnyvale, CA. Bohdan Balko, Chairman

Descriptors, Keywords: Bayesian Network Midcourse Multi-target Tracking

Pages: 00050

Cataloged Date: Dec 05, 1994

Document Type: HC

Number of Copies In Library: 000001

Record ID: 29571

INTRODUCTION

- BAYESIAN NETWORKS (INFLUENCE DIAGRAMS) HAVE EVOLVED OVER THE LAST DECADE INTO A POWERFUL TOOL FOR PROBABILISTIC INFERENCE:
  - HOWARD & MATHESON (1981) (SEMINAL PAPER)
  - SCHACTER (1986) (DISCRETE INFLUENCE DIAGRAMS)
  - KENLEY (1986) (NORMAL INFLUENCE DIAGRAMS)
  - PEARL (1986) (BAYESIAN NETWORKS)
- INFLUENCE DIAGRAMS PROVIDE A FRAMEWORK TO REPRESENT AND MANIPULATE JOINT PROBABILITY DISTRIBUTIONS FOR COMPLEX NETWORKS OF RANDOM VARIABLES
- INFLUENCE DIAGRAMS CAN BE USED TO IMPLEMENT WITHIN THE SAME FRAMEWORK
  - STATE ESTIMATION (LINEAR GAUSSIAN) } TRACKING
  - DATA ASSOCIATION (DISCRETE) } TRACKING
  - TRACK PROMOTION (DISCRETE)
- LMSC HAS BUILT A LIBRARY OF INFLUENCE DIAGRAM UTILITIES TO PROTOTYPE MIDCOURSE TRACKING ALGORITHMS
  - ALGORITHM PERFORMANCE
  - THROUGHPUT/MEMORY (NONOPTIMIZED)

K9-7246/061

INTRODUCTION

This presentation discusses the application of Bayesian Networks or Influence Diagrams to the implementation of midcourse tracking algorithms. The Influence Diagram is used to represent and manipulate probabilistic information in complex networks of random variables. The generic capabilities of the Influence Diagram are used to carry out the major tracking functions, including linear gaussian state estimation, data association hypothesis scoring and track promotion scoring.

LMSC has built a library of Influence Diagram utilities to construct, scan and manipulate an Influence Diagram. These utilities are used in implementing the midcourse tracking algorithm in order to assess algorithm performance and to begin to estimate throughput and memory requirements. The throughput and memory requirements are upper bound estimates at this stage since the algorithms are executing within a generic environment which is not tailored and optimized for a specific hardware environment.

AGENDA

- INFLUENCE DIAGRAMS USED TO REPRESENT UNCERTAIN KNOWLEDGE IN COMPLEX SYSTEMS
  - GENERIC REPRESENTATION
  - APPLICATION TO MIDCOURSE TRACKING
- OPERATIONS ON INFLUENCE DIAGRAMS TO PERFORM INFERENCING
  - GENERIC OPERATIONS
  - APPLICATION TO MIDCOURSE TRACKING

K9-7246/033

AGENDA

The agenda will cover both the generic aspects of Influence Diagrams and their application to the midcourse tracking problem. Furthermore, the agenda will be partitioned into a discussion of the capability of the Influence Diagram to represent uncertain or probabilistic information in complex systems, and the operations used in manipulating the Influence Diagram.

BAYESIAN NETWORK/INFLUENCE DIAGRAM DEFINED

- ACYCLIC DIRECTED GRAPH REPRESENTING THE JOINT PROBABILITY DISTRIBUTION FOR A SET OF RANDOM VARIABLES
  - NODES = RANDOM VARIABLE
  - ARC = PROBABILISTIC CONDITONING

EXAMPLES

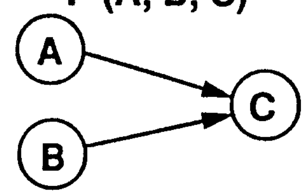
P (A, B)

P (A) P (B/A)

P (A, B)

P (B) P (A/B)

P (A, B, C)



P (A) P (B) P (C/A,B)

- OBSERVATIONS
  - A JOINT PROBABILITY DISTRIBUTION CAN BE REPRESENTED BY MANY INFLUENCE DIAGRAMS - ONE FOR EACH DECOMPOSITION.
  - LACK OF AN ARC REPRESENTS CONDITIONAL INDEPENDENCE.

K9-7246/002

BAYESIAN NETWORK/INFLUENCE DIAGRAM DEFINED

This chart defines the Influence Diagram as a device to represent a joint probability distribution of a set of random variables. Each random variable is represented as a node and conditional dependence between random variables is represented as an arc between the corresponding nodes.

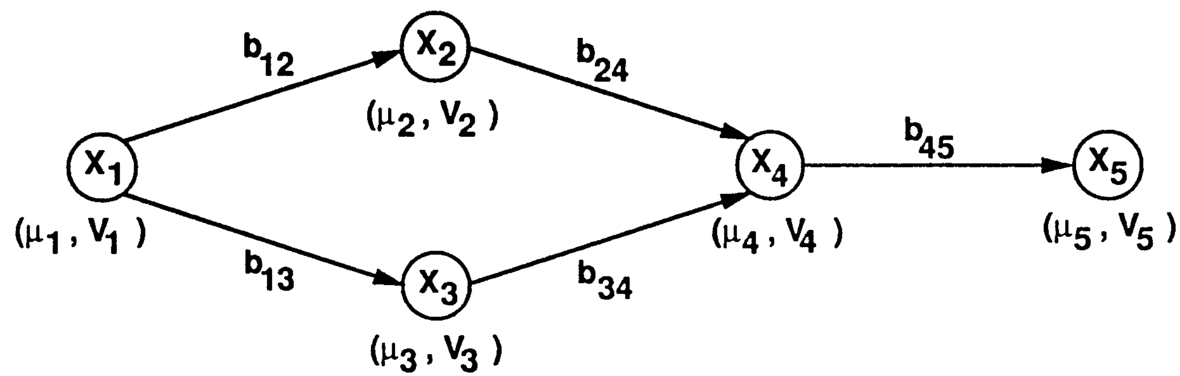
A joint probability distribution can be factored in many ways. In the example, the joint distribution, P(A,B) can be written as P(A)*P(B/A) or as P(B)*P(A/B). Each factorization or decomposition, is represented by a specific Influence Diagram.

The lack of an arc between two nodes indicates that the corresponding random variables are conditionally independent of each other.

09/07/1989 11:30 LMSC PANAFAX UF-600 **** 14711290 P.04

NORMAL INFLUENCE DIAGRAM

SCALAR NODE EXAMPLE



- $N = \{1, 2, \ldots, n\}$
- $X_N = (X_1, \ldots, X_n)$ X_i IS A SCALAR NORMAL RANDOM VARIABLE

  $E[X_N] = \mu_N$ (n x 1)

  $\mathrm{Cov}[X_N] = \Sigma_{NN}$ (n x n)
- CONDITIONAL DISTRIBUTIONS

  $E[X_j \mid X_{c(j)} = \chi_{c(j)}] = \mu_j + \sum_{k \varepsilon C(j)} b_{kj} (\chi_k - \mu_k)$

  $\mathrm{Var}[X_j \mid X_{c(j)} = \chi_{c(j)}] = v_j$

  C (j) = CONDITIONAL PREDECESSORS OF NODE j

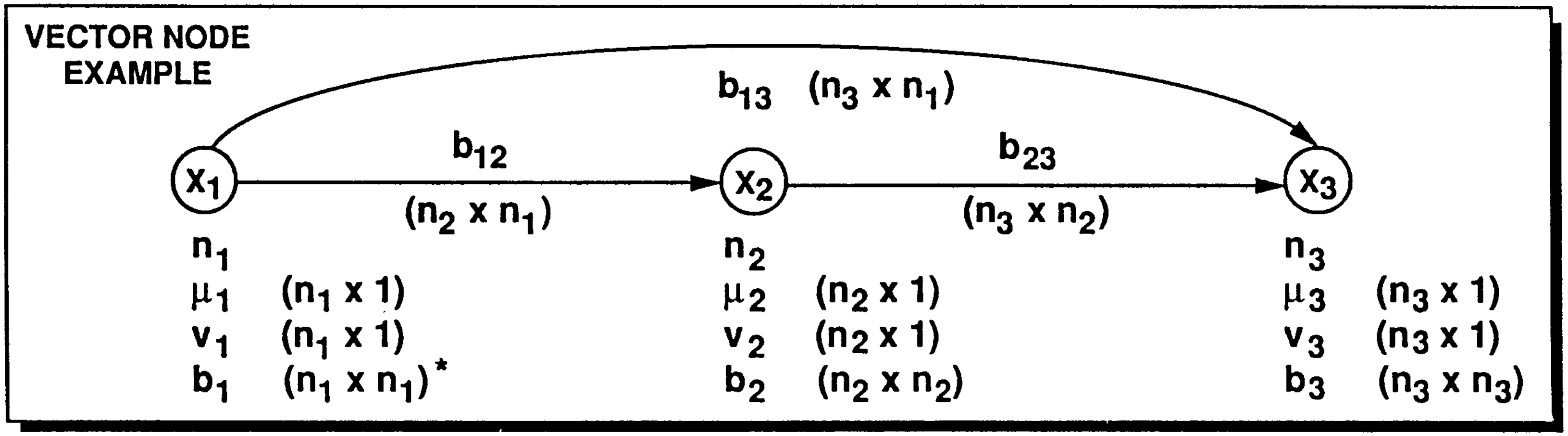
K9-7246/003

NORMAL INFLUENCE DIAGRAM

This chart presents an example of an Influence Diagram for the joint probability density for a set of 5 normal random variables.

Each random variable is a scalar random variable with an unconditional mean, mu, and conditional variance, v. The arc strengths, bij, represents the influence of the ith random variable on the jth random variable and are used in the expression for the conditional mean of jth random variable.

NORMAL INFLUENCE DIAGRAM



- $N = \{1, 2, \ldots, n\}$

- $X_N = (X_1, \ldots, X_n)$ X_i IS A VECTOR NORMAL RANDOM VARIABLE OF LENGTH N_i

  $E[X_N] = \mu_N$ $(m \times 1)$ $m = \sum_i n_i$

  $COV[X_N] = \Sigma_{NN}$ $(m \times m)$

- CONDITIONAL DISTRIBUTIONS

  $$E[X_j / X_{c(j)} = x_{c(j)}] = \mu_j + \sum_{K \varepsilon c(j)} b_{kj}(x_k - \mu_k)$$

*NOTE: b_i HAS $n_i \times (n_i - 1)$ 12 INDEPENDENT COMPONENTS

K9-7246/004

NORMAL INFLUENCE DIAGRAM

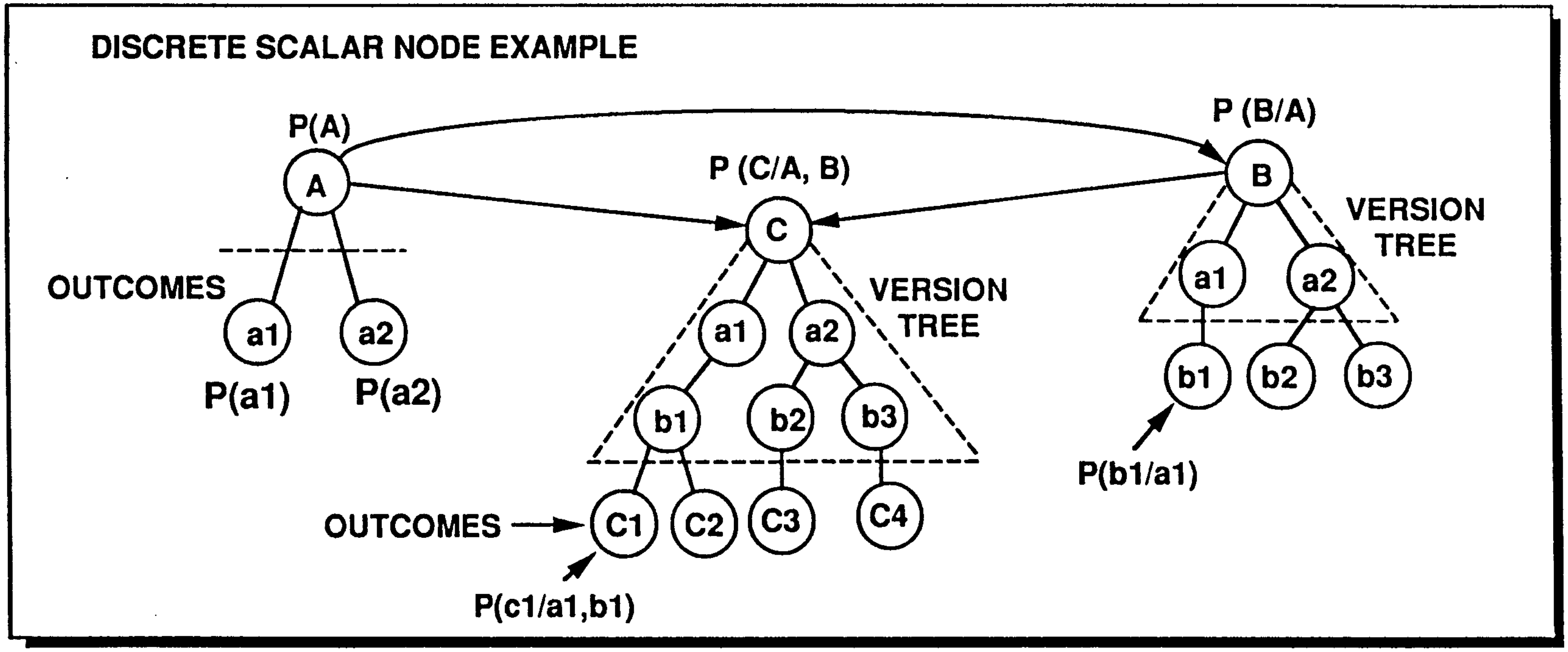
This chart presents the influence diagram for 3 normal random variables in which each normal random variable is a vector.

Each vector variable is represented by the unconditional mean vector (mui), the conditional variance vector (vi) and the internal arc strengths (bi).

The conditional dependence between the vector variables is represented by the external arc strengths (bij).

It should be noted that the internal arc strengths (bi) have ni*(ni-1)/2 components

DISCRETE INFLUENCE DIAGRAM



- JOINT DISCRETE RANDOM VARIABLE S = (A, B, C)
  A, B, C ARE DISCRETE RANDOM VARIABLES
- CONDITIONAL PROBABILITIES
  P(A) P(B/A) P(C/A, B)

K9-7246/015

DISCRETE INFLUENCE DIAGRAM

This chart shows an example of an Influence Diagram for 3 discrete random variable, (A,B,C). The joint probability density, P(A,B,C), is factored into P(A)*P(B/A)*P(C/A,B) which is represented in the diagram.

Variable A has two outcomes: (a1,a2) The associated probabilities are P(a1) and P(a2).

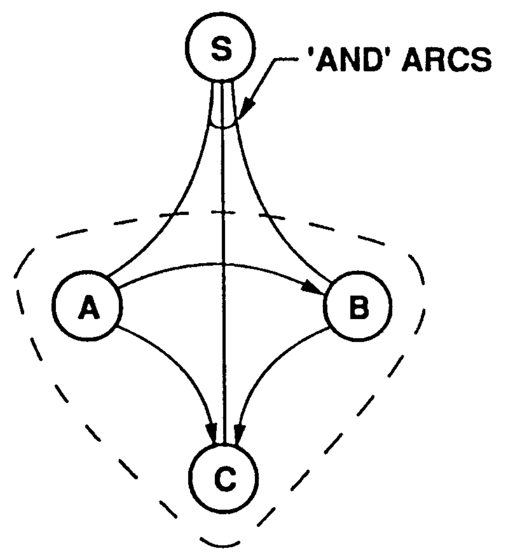
These two outcomes for A become versions for variables B and C. Assuming A=a1, then B has the single outcome b1 and conditional probability P(b1/a1) which would equal 1.0. Assuming A=a2, then B has two outcomes: (b2,b3). The associated probabilities are p(b2/a2) and p(b3/a2).

Variable C has a 2 level version tree, that is, an outcome from both A and B must be specified before the outcomes for C can be defined. For example, for A=a1 and B=b1, C has two outcomes: (c1,c2). Their probabilities are P(c1/a1,b1) and P(c2/a1,b1). For A=a2 and B=b2, C has one outcome: c3, and for A = a2 and B=b3, C has one outcome: c4.

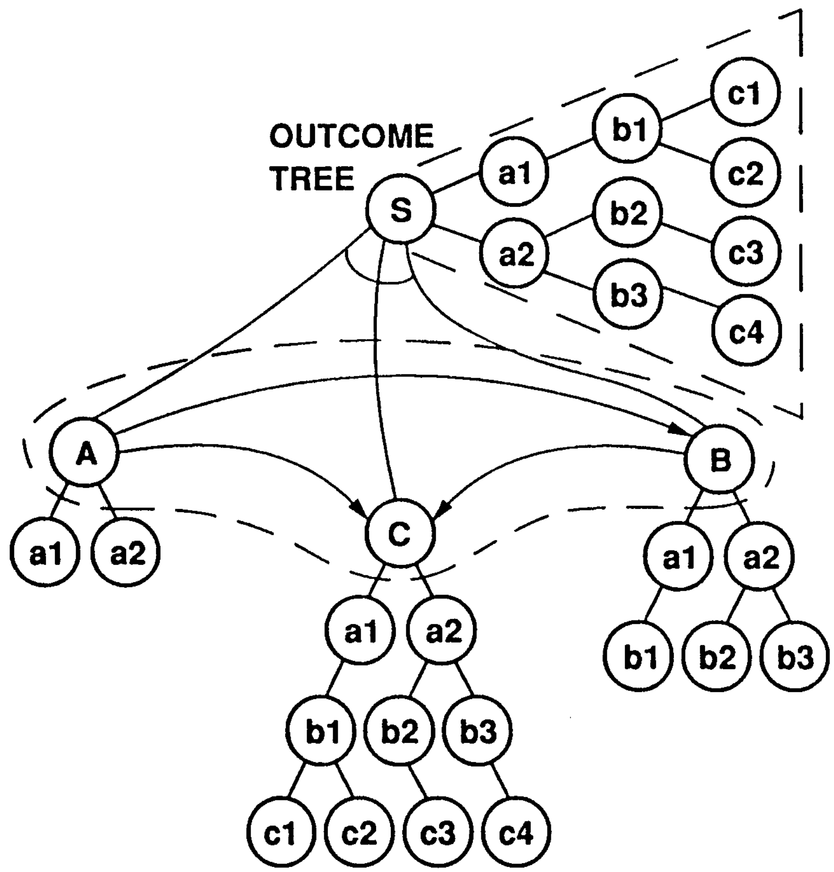
It should be noted that the Influence Diagram is represented by the three root nodes (A,B,C) and the directed arcs. The other nodes represent data internal to the root nodes such as outcomes and versions.

DISCRETE INFLUENCE DIAGRAM

DISCRETE VECTOR NODE EXAMPLE



S IS UNELABORATED



S IS FULLY ELABORATED

- S = {A, B, C} IS A VECTOR DISCRETE NODE

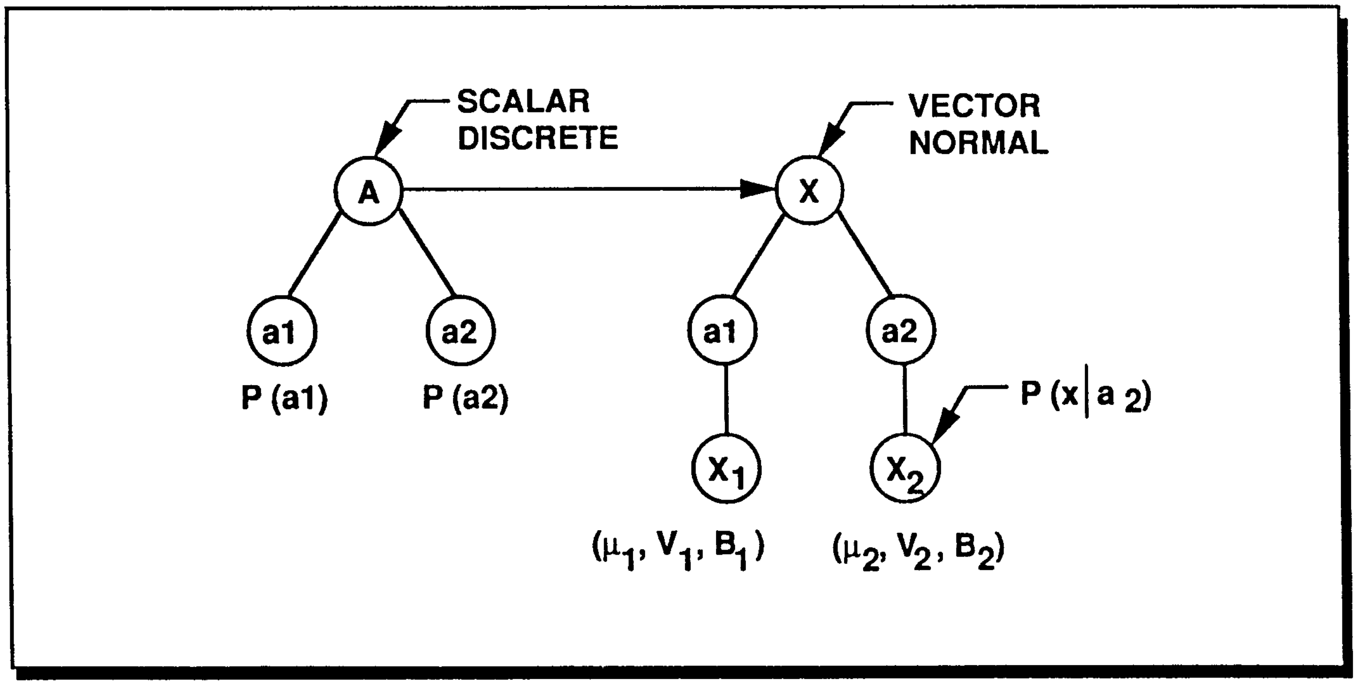
K9-7246/042

DISCRETE INFLUENCE DIAGRAM

This chart shows an example of a single discrete vector node. The vector node represents the joint random variable for a set of three scalar nodes, (A,B,C). The outcomes for S are joint outcomes for the three scalar nodes.

On the left side of the diagram, S is shown connected by 'and' arcs to the three nodes. The 'and' arcs indicate that S consists of A and B and C. The right side of the diagram shows the S node fully elaborated with the joint outcomes froma A, B and C.

MIXED DISCRETE PLUS NORMAL INFLUENCE DIAGRAM



- THE INFLUENCE OF THE DISCRETE RANDOM VARIABLE "SPLITS" THE NORMAL RANDOM VARIABLE INTO VERSIONS ("OR" SPLITS)

K9-7246/041

MIXED DISCRETE PLUS NORMAL INFLUENCE DIAGRAM

This chart shows an example of a scalar discrete node influencing a vector normal random variable. Each outcome of A becomes a version for X. Each version for X has an unconditional mean vector, conditional variance vector and internal arc strengths.

The discrete node can be viewed as splitting the normal node into versions. This splitting is termed an 'or' split to indicate that the X node is X1 (under A=a1) or X2 (under A=a2).

AGENDA

- INFLUENCE DIAGRAMS USED TO REPRESENT UNCERTAIN KNOWLEDGE IN COMPLEX SYSTEMS
  - GENERIC REPRESENTATION
  - APPLICATION TO MIDCOURSE TRACKING
- OPERATIONS ON INFLUENCE DIAGRAMS TO PERFORM INFERENCING
  - GENERIC OPERATIONS
  - APPLICATION TO MIDCOURSE TRACKING

K9-7246/032

INFLUENCE DIAGRAM NODES USED IN MIDCOURSE TRACKING

NODE	NAME	TYPE	SIZE	OUTCOMES
Z	FOCAL PLANE CONTACT	CTNS*	VECTOR (2 x 1)	(AZ, EL, INTENSITY, EXTENT (3))
F	FOCAL PLANE TRACK	CTNS	VECTOR (6 x 1)	$(AZ, \dot{AZ}, \ddot{AZ}, EL, \dot{EL}, \ddot{EL})$
X	CARTESIAN TRACK	CTNS	VECTOR (6 x 1)	$(X, Y, Z, \dot{X}, \dot{Y}, \dot{Z})$
E	EXTENT (ELLIPSE/ELLIPSOID)	CTNS	VECTOR (3 x 1/6 x 1)	$(\delta AZ, \delta AZEL, \delta EL)/(\delta X, \delta XY, ..., \delta Z)$
C	CONTACT ASSIGNMENT	DISCRETE	SCALAR	CONTACT TO TRK ASSIGNMENT | NEW TRACK | FALSE ALARM
t^+	TRACK UPDATE ASSIGNMENT	DISCRETE	SCALAR/ VECTOR	TRK TO CONTACT ASSIGNMENT | MISSED DETECTION
t^-	TRACK PREDICTION ASSIGNMENT	DISCRETE	SCALAR/ VECTOR	NO SPLIT | SPLIT INTO 2 | SPLIT INTO 3, . . .
S	SCENE	DISCRETE	VECTOR	JOINT SET OF CONTACT, UPDATE AND PREDICTION ASSIGNMENTS

*ALL CONTINUOUS NODES ARE ASSUMED NORMAL

K9-7246/008

INFLUENCE DIAGRAM NODES USED IN MIDCOURSE TRACKING

This chart shows the nodes used in the midcourse tracking algorithm at this stage of development. The char shows the node symbol, the node name, its type (continuous or discrete), its size (scalar, discrete or both) and the outcomes for the node.

The focal plane contact is the random variable representing the estimated line of sight for a target or false alarm. (Note that the outcome column erroneously includes the intensity and extent.)

The focal plane track is the 6 element state vector for a track on the focal plane.

The cartesian track is the 6 element state vector for a track existing in 3-d space.

The extent can be either 3 state or 6 state and represents the parameters of an ellipse or ellipsoid, respectively of a cluster of objects.

The contact assignment variable is a discrete random variable that identifies the possible assignment outcomes for a contact. A contact can be an update to an existing track, the start of a new track or a false alarm.

The track update assignment random variable is a discrete random variable that identifies the possible assignment outcomes for a track. A track can be updated by one or more contacts or not be updated ar all (missed detection).

Thhe track prediction assignment random variable is a discrete random variable that identifies the possible dynamical models for the track. The models entertained for ballistic targets are that the track does not split or splits into two tracks or splits into 3 tracks or etc. These are 'and' splits.

The scene is a vector discrete random variable that represents the joint outcomes of a set of c, t+ and t- nodes.

NORMAL RANDOM VARIABLES USED IN MIDCOURSE TRACKING

CONTACT LINE OF SIGHT	FOCAL PLANE TRACK	CARTESIAN TRACK	FOCAL PLANE EXTENT
Z $\mu = (az, el)$ V (2x1) b (2x3)/2	F $\mu = (az, \dot{az}, \ddot{az}, el, \dot{el}, \ddot{el})$ V (6x1) b (6x7)/2	X $\mu = (x, y, z, \dot{x}, \dot{y}, \dot{z})$ V (6x1) b (6x7)/2	E_3 $\mu = (\delta az, \delta azel, \delta el)$ V (3x1) b (3x4)/2

CARTESIAN EXTENT	CONTACT STATE	TRACK STATE
E_6 $\mu = (\delta x, \delta xy, \ldots . \delta z)$ V (6x1) b (6x7)/2	Z — E_3	a. F — E_3 b. X — E_3 c. X — E_6

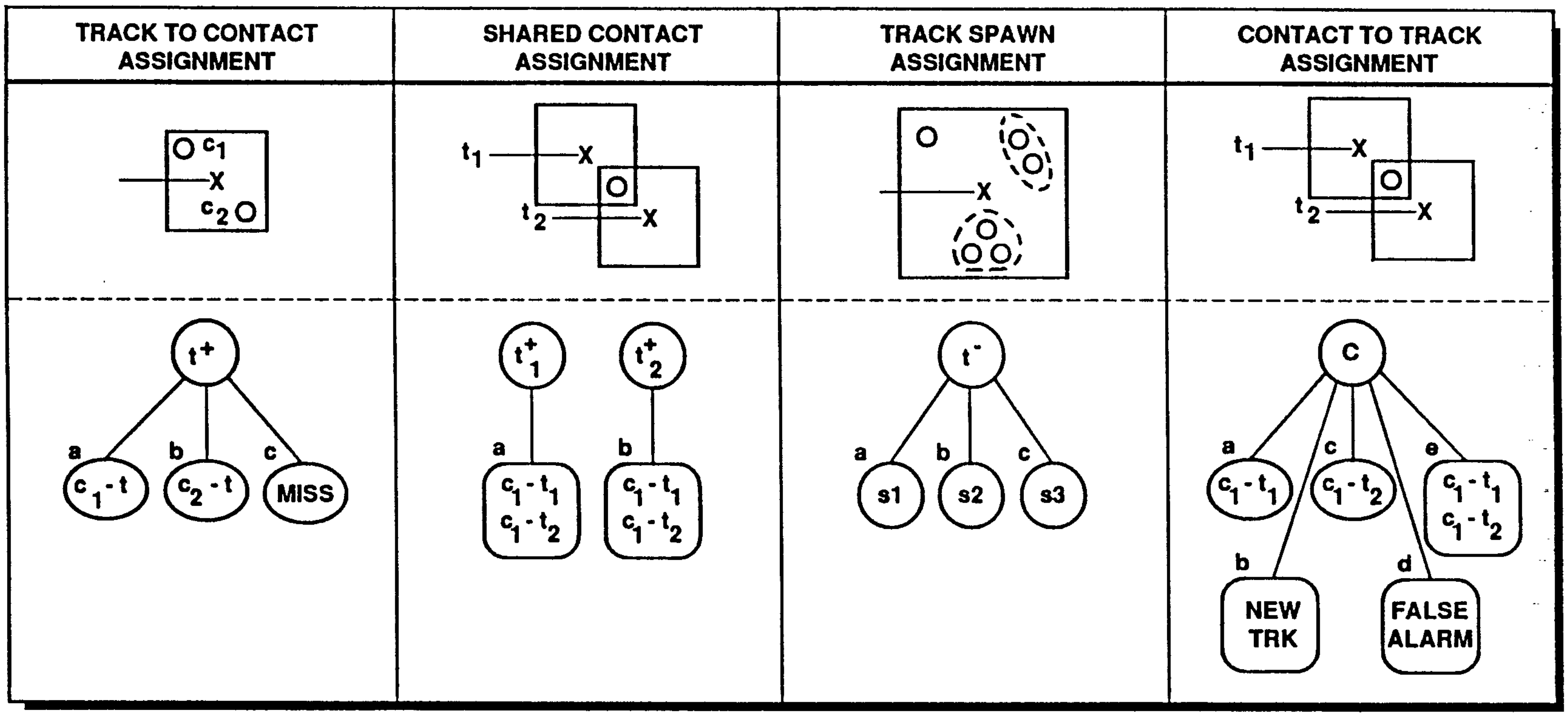
K9-7246/040

NORMAL RANDOM VARIABLES USED IN MIDCOURSE TRACKING

This chart details the normal random variables

The contact state combines the z and E3 node. The track state combines the F and E3, X and E3 or X and E6 nodes.

DISCRETE RANDOM VARIABLES USED IN MIDCOURSE TRACKING (1 OF 2)



a. ASSIGN c_1 TO t
b. ASSIGN c_2 TO t
c. ASSIGN MISS TO t

a. ASSIGN $_1$c TO $_1$t AND$_2$ t
b. ASSIGN $_1$c TO $_1$t AND$_2$ t

a. DO NOT SPAWN TRACK
b. SPAWN TRACK INTO TWO TRACKS
c. SPAWN TRACK INTO THREE TRACKS

a. ASSIGN c_1 TO t_1
b. c_1 IS A NEW TRACK
c. ASSIGN c_1 TO t_2
d. c_1 IS A FALSE ALARM
e. ASSIGN c_1 TO t_1 AND t_2

K9-7246/039

DISCRETE RANDOM VARIABLES USED IN MIDCOURSE TRACKING (1 OF 2)

This chart shows various examples for the discrete random variables.

The first column shows the case of two contacts in a track gate. The update assignment node, t+, has three possible outcomes: c1 is assigned to the track (c1-t), c2 is assigned to the track (c2-t) or the track has a miss (miss).

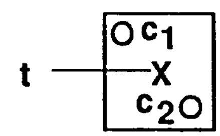
The second column shows the special case of a shared contact. A single contact lies in the overlap region of two tracks and it has been determined that the contact should be assigned to both tracks at the same time. The shared outcome is represented as a set of contact to track assignements: c1-t1 and c1-t2.

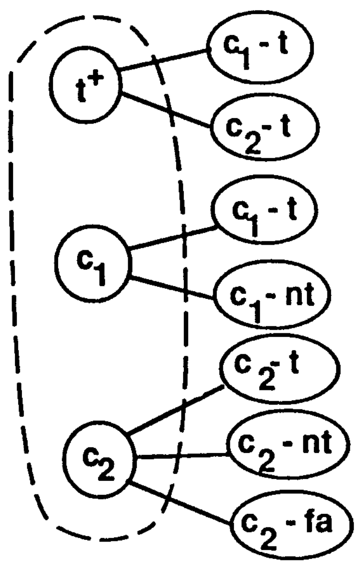
The third column shows the case of track spawning ('and' splits). The example shows that the track can split into 3 tracks or into 2 track or not split at all.

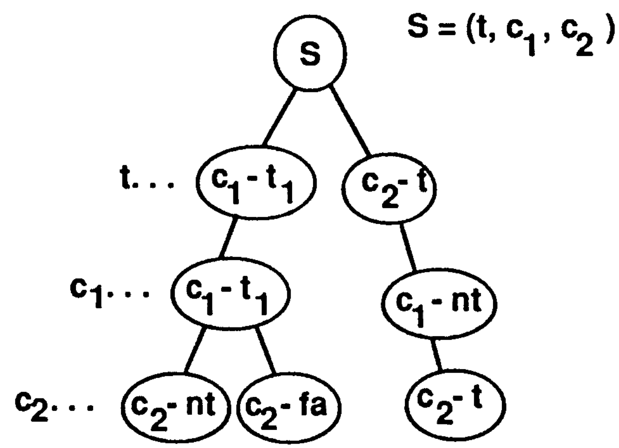
The fourth column shows the case of a contact falling into the overlap region of two track gates. The contact assignment node has 5 possible outcomes: assign the contact to t1 only, assign the contact to t2 only, assign the contact to t1 and t2, consider the contact to be the start of a new track or consider the contact to be a false alarm.

DISCRETE RANDOM VARIABLES USED IN MIDCOURSE TRACKING (2 OF 2)

SCENE - JOINT SET OF ASSIGNMENTS







3 JOINT OUTCOMES

K9-7246/038

DISCRETE RANDOM VARIABLES USED IN MIDCOURSE TRACKING (2 OF 2)

This chart shows the outcomes for the scene node for the case of two contacts in the gate for a single track. The s-node represents the joint probability of the t+ node and the two c-nodes.

In this example, three joint outcomes are feasible. The first one assigns c1 to the track and declares that c2 is a new track. The second outcome assigns c1 to the track and declares that c2 is a false alarm. The third outcome assigns c2 to the track and declares that c1 is a new track.

EXAMPLES OF TRACKS (1 OF 4)

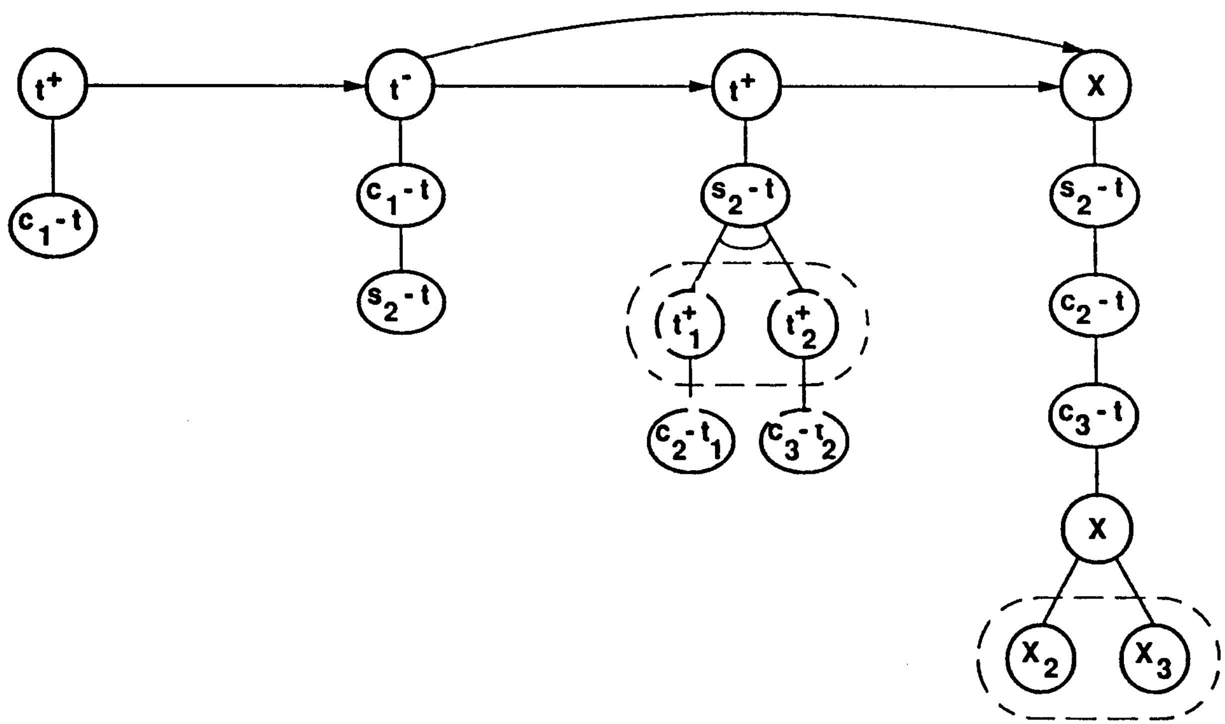
The next set of 4 charts show examples of the nodes making up various tracks in the case of a single satellite.

The first track is a new track. A c-node is given a new track and false alarm outcome which influences the track's first t+ node. The t+ considers the contact assignment outcome for the new track version and false track outcome for the false alarm version. The t+ node influences the F node which is split.

The second track is an urambiguous track with no spawn or update ambiguity. The first t+ node, which is from the previous frame, has a single contact assignment. The t- node has one outcome for the single assignment. The t+ node for the current frame has a single contact update.

The third track has an update ambiguity on the current frame. The track can be updated by one of two contacts. The X-node is split into two versions.

EXAMPLES OF TRACKS (2 OF 4)



TRACK

- NO SPAWN AMBIGUITY
- NO UPDATE AMBIGUITY

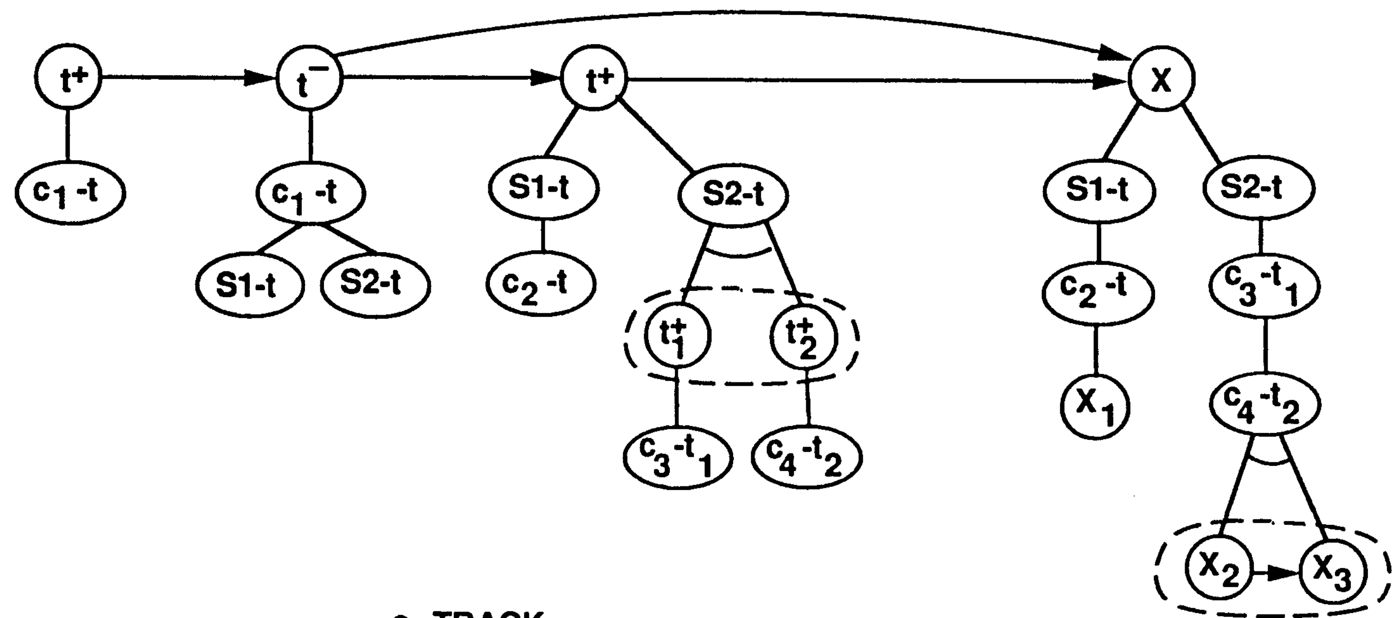
K9-7246/046

EXAMPLES OF TRACKS (2 OF 4)

This chart shows a track which has no ambiguity but shows how a spawn is handled.

The t- node has a single outcome which specifies that the track should be split into two tracks. As a result the t+ node generates a vector version, represented by the 'and' arcs coming out of the version node, s2-t. The vector version consists of two scalar t+ nodes one for each split track. In this case each scalar node has a single update assignment. The X-node is shown with the two track formation shown. (It should be noted that there should be an 'and' arc connecting the lines connecting the X-node with X2 and X3.)

EXAMPLES OF TRACKS (3 OF 4)



- TRACK
  - SPAWN AMBIGUITY
  - NO UPDATE AMBIGUITY

K9-7246/045

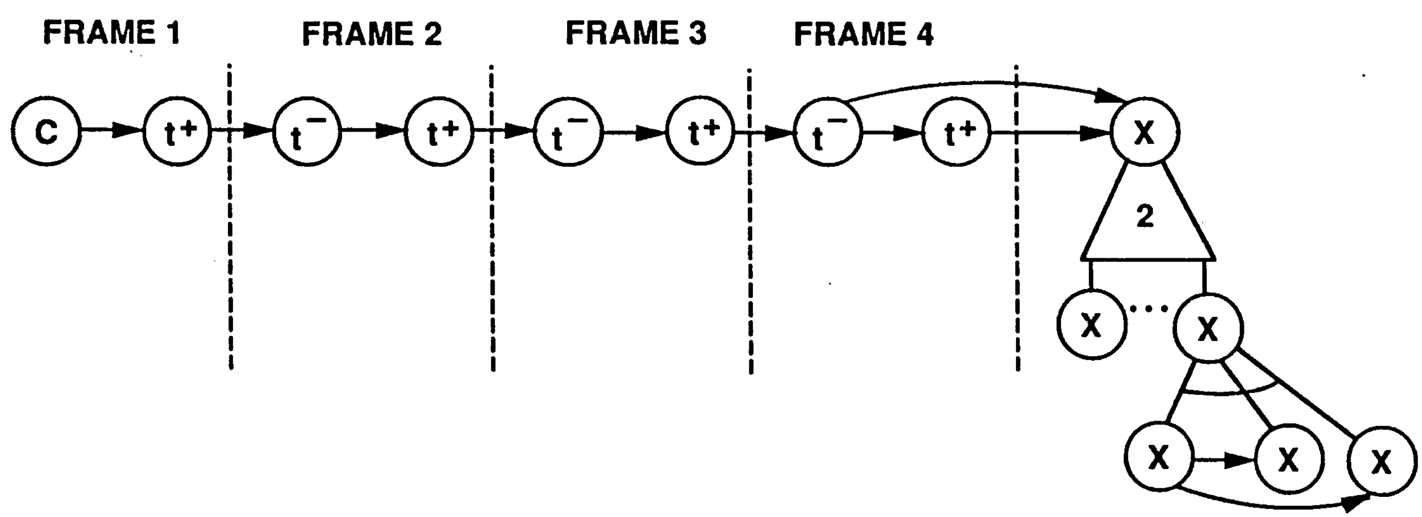
EXAMPLES OF TRACKS (3 OF 4)

This chart shows a track with a spawn ambiguity bu t no update ambiguity.

The t- node has two spawn outcomes: s1 and s2. The s1 outcome specifies that no spawning should take place. The s2 outcome specifies that the track should be split into two tracks thereby creating a two track formation. Each version of the t+ node has unambiguous update assignments leading to the X-node shown.

EXAMPLES OF TRACKS (4 OF 4)

COMPLETE TRACK HISTORY



- A MOVING WINDOW OF THE t NODE CHAIN IS MAINTAINED
  - WHEN THE FIRST NODE IN THE CHAIN HAS A SINGLE OUTCOME, IT IS DELETED AND THE CHAIN IS 'CLEANED UP'

K9-7246/044

EXAMPLES OF TRACKS (4 OF 4)

This chart shows the general configuration of discrete nodes and the continuous node for a track. If the c-node and t-nodes were not cleaned up, there would exist a chain of nodes beginning with the c-node and initial t+ node and continuing with a t- and t+ node pair on each frame. The t- and t+ nodes on the last frame influence the continuous node. The continuous node has a two level version tree with singleton track or formation track versions.

The chain is not allowed to grow indefinitely. As data is received, the outcome probabilities for the first node in the chain are updated and decisions are made to prune or select an outcome. When a single outcome remains for the first node in the chain, it is deleted and the chain is cleaned up. In this way, a moving window of t nodes is maintained.

AGENDA

- INFLUENCE DIAGRAMS USED TO REPRESENT UNCERTAIN KNOWLEDGE IN COMPLEX SYSTEMS
  - GENERIC REPRESENTATION
  - APPLICATION TO MIDCOURSE TRACKING
- OPERATIONS ON INFLUENCE DIAGRAMS TO PERFORM INFERENCING
  - GENERIC OPERATIONS
  - APPLICATION TO MIDCOURSE TRACKING

K9-7246/034

INFLUENCE DIAGRAM OPERATIONS

CONSTRUCTORS (1)	SELECTORS (2)		ITERATORS (3)	ERROR CONDITIONS
INITIALIZE	IS-EMPTY-THE-DIAGRAM	GET-OUTCOMES	DEPTH-FIRST-SEARCH	VERTEX-NUMBER-OVERFLOW
ADD-VERTEX	ITEM-OF-VERTEX	VERSIONS-OF	BREADTH-FIRST-SEARCH	VERTEX-IS-NULL
DELETE-VERTEX	IS-DISCRETE	IS-VERSION	LOCATION-OF	VERTEX-IS-NOT-IN-GRAPH
DUPLICATE-VERTEX	IS-CONTINUOUS	VERSION-BOUNDRY	FIND-PATH	VERTEX-HAS-REFERENCES
REPLACE	IS-NULL-THE-VERTEX	VERSION-PART	FIND-THE-ARC	ARC-IS-NULL
COPY-VERTEX	IS-A-MEMBER	TOP-VERSION	VISIT-VERSIONS	ARC-IS-NOT-IN-GRAPH
SET-ITEM-OF-VERTEX	ATTRIBUTE-OF-ARC	BOTTOM-VERSION	VISIT-VERTICES	ARC-IS-RELEVANT
ADD-OUTCOME	NUMBER-ARCS-FROM	DIRECT-PREDECESSORS	VISIT-ARCS	GRAPH-HAS-CIRCUIT
REMOVE-OUTCOME	SOURCE-OF	IS-PREDECESSOR	PARENT-OF	ITEM-NOT-DEALLOCATED
SINGLE-OUTCOME-CLEANUP	DESTINATION-OF	IS-SUCCESSOR	SUBTREE-OF	IS-NOT-REVERSIBLE
ADD-ARC	IS-NULL-THE-ARC	DIRECT-SUCCESSORS	TREE-OF	IS-NOT-A-ROOT-VERTEX
DELETE-NON-RELEVANT-ARC	IS-OUT-ARCS	WEAK-PREDECESSORS	ROOT-OF	IS-NOT-A-DISCRETE-VERTEX
DESTROY-ARC	IS-IN-ARCS	COMMON-PREDECESSORS		IS-NOT-A-CONTINUOUS-VERTEX
REVERSE ARC	IS-REVERSIBLE	IS-ROOT		IS-NOT-A-OUTCOME-VERTEX
REMOVE ARC	IS RELEVANT	IS-LEAF		IS-NOT-VALID-VERTEX-LABEL
SET-ATTRIBUTE-OF-ARC	ARC-EXISTS-BETWEEN	GET-LEAFS		PATH-NOT-FOUND
DESTROY-TREE	OUTCOME-PART	GET-PATH		BAD-SATELLITE-NUMBER
COPY-TREE	TOP-OUTCOME	LEAF-VERSIONS-OF		BAD-VERTEX-NUMBER
PRUNE-PATH	BOTTOM-OUTCOME	CHILDREN-OF		
PROPAGATE	IS-OUTCOME	SIZE-OF		
INSTANTIATE	IS-EQUAL	ROOT-OF		
NORMALIZE	GET-UNIQUE-OUTCOMES	RANDOM-VARIABLE-OF		
SEQUENCE				
PROJECT				
INCORPORATE				
INFER				

(1) CONSTRUCTORS ALTER THE STATE OF THE INFLUENCE DIAGRAM
(2) SELECTORS EVALUATE THE CURRENT STATE OF THE INFLUENCE DIAGRAM
(3) ITERATORS VISIT DIFFERENT PARTS OF THE INFLUENCE DIAGRAM

K9-7246/013

INFLUENCE DIAGRAM OPERATIONS

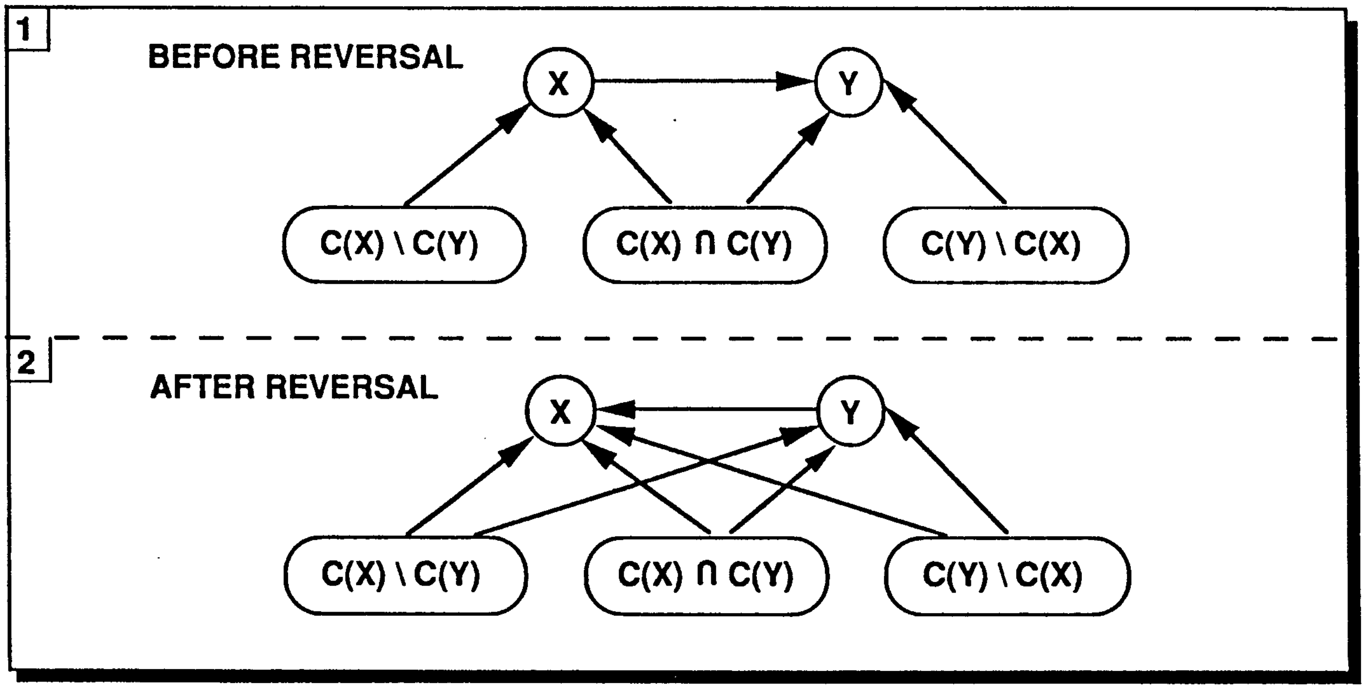
This chart shows the set of utilities created to construct, evaluate and scan the Influence Diagram. These utilities represent about 2 manyears of effort and are written in ADA.

The main routines used in the midcourse tracking algorithm are Reverse_Arc, Instantiate, Infer and Project. These will be discussed in the following charts.

The utility, Is_Relevent, should be mentioned. The routine examines arcs between continuous node to determine if their arc strengths are strong enough to be maintained., Thus, a tradeoff between the processing cost of maintaining arcs and the reduction in performance by deleting arcs can be attained.

REVERSE ARC

- REVERSE ARC IS THE INFLUENCE DIAGRAMS INPLEMENTATION OF BAYES' RULE



- EACH NODE INHERITS THE PREDECESSORS OF THE OTHER NODE

K9-7246/037

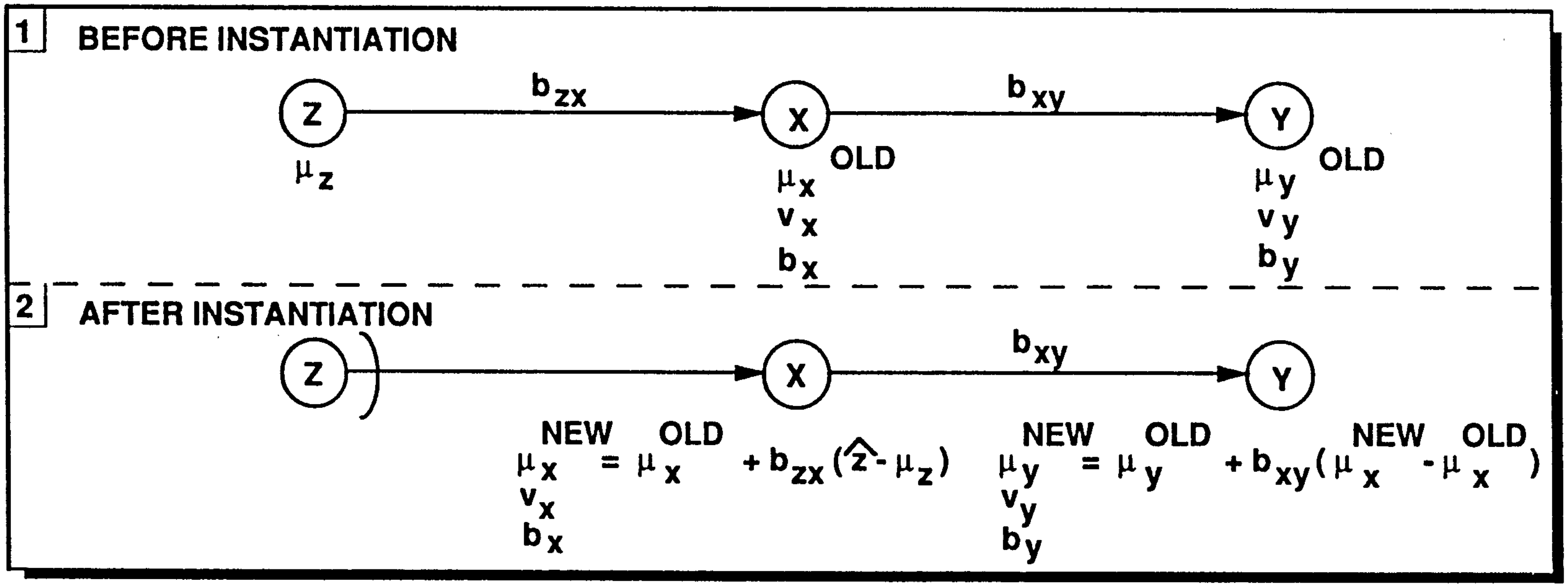
REVERSE ARC

Reverse_Arc is an important utility since it carries out Bayes' rule. In carrying out the arc reversal, each node inherits the predecessors of the other node. Reversing the same arc twice does not get back to the same diagram unless the arcs are tested for relevancy and the irrelevent arcs deleted.

Note that C(X) represents the set of conditional predecessors for the node X.

INSTANTIATE

- **INSTANTIATE** UPDATES THE DIAGRAM AS A RESULT OF A MEASUREMENT OF A RANDOM VARIABLE



- THE INSTANTIATE ACTION FLOWS THROUGH THE DIAGRAM UPDATING THE CONDITIONAL MEAN OF THE SUCCESSORS, THE SUCCESSORS OF THE SUCCESSORS, ETC. OF THE INSTANTIATED VARIABLE
- AFTER INSTANTIATION, THE INSTANTIATED VARIABLE CAN BE DELETED

K9-7246/036

INSTANTIATE

Instantiate is the utility which incorporates a measured value for a continuous random variable. In the chart, diagram is shown before and after the variable z is instantiated.

When z is instantiated with the measured value, the unconditional mean of X is updated. Furthermore, because the unconditional mean of X is update then the unconditional mean of Y is also updated. The other parameters in the diagram are unchanged.

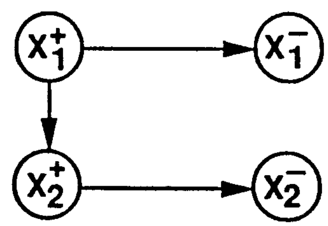
After z is instantiaed, it can be deleted since it is no longer a random variable.

INFER

- INFER IS AN ORDERED SEQUENCE OF ARC REVERSALS TO STRUCTURE THE INFLUENCE DIAGRAM TO REPRESENT A DESIRED CONDITIONAL PROBABILITY

1

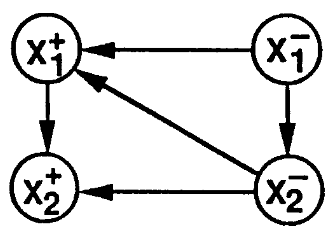
BEFORE INFERENCE OF { X_1^- , X_2^- } ONTO { X_1^+ , X_2^+ }



DESIRE P [X_1^+ , X_2^+ / X_1^- , X_2^-]

2

AFTER INFERENCE OF { X_1^- , X_2^- } ONTO { X_1^+ , X_2^+ }



K9-7246/043

INFER

Infer is a major function used in the midcourse tracking algorithm. It carries out an order sequence of arc reversals to structure the Influence Diagram into a desired form.

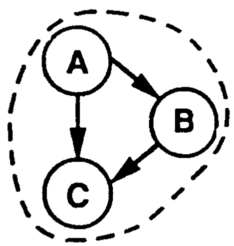
In the example, it is desired that the set {X1-, X2-} influence {X1+, X2+}. The Infer utility carries out an ordered sequence of reversals. The order to determined to assure that no loops exist in the diagram.

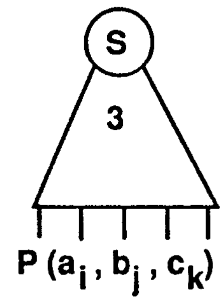
The diagram is shown after Inference. Note that an additional arc from X2- to X1+ is generated.

PROJECT

- <u>PROJECT</u> CALCULATES THE PROBABILITIES OF THE OUTCOMES OF RANDOM VARIABLES IN A SUBDIAGRAM GIVEN THE PROBABILITIES OF THE JOINT OUTCOMES OF THE SUBDIAGRAM

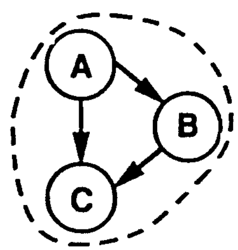
1 BEFORE PROJECTION ONTO A



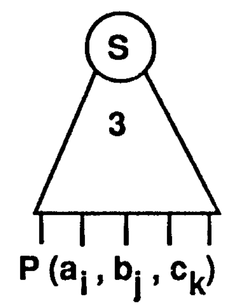


3 LEVEL
OUTCOME TREE

2 AFTER PROJECTION ONTO A



$$P\ (a_i) = \sum_{c_k} \sum_{b_j} P\ (a_i, b_j, c_k)$$



- PROJECT 'SUMS OUT' THE UNWANTED OUTCOMES IN THE JOINT RANDOM VARIABLE

K9-7246/035

PROJECT

Project is the utility that takes the joint outcome probabilities for a set of random variablesand calculates the marginal probabilities for a given random variable in the set.

In the example, the variable S represents the variables {A,B,C}. It is assumed that the joint outcome probabilities exist, P(ai,bj,ck). Next, the marginal probability p(ai) is desired. The Project operator carries out the summation as shown in the chart to calculate P(ai).

09/07/1989 11:39 LMSC PANAFAX UF-600 **** 14711290 P.22

AGENDA

- INFLUENCE DIAGRAMS USED TO REPRESENT UNCERTAIN KNOWLEDGE IN COMPLEX SYSTEMS
  - GENERIC REPRESENTATION
  - APPLICATION TO MIDCOURSE TRACKING
- OPERATIONS ON INFLUENCE DIAGRAMS TO PERFORM INFERENCING
  - GENERIC OPERATIONS
  - APPLICATION TO MIDCOURSE TRACKING

K9-7246/031

INFLUENCE DIAGRAM OPERATIONS APPLIED TO MIDCOURSE TRACKING

- STATE ESTIMATION
  - KALMAN FILTER PROCESSING
  - FORMATION TRACK UPDATE
  - TRACK SPAWNING ('AND' SPLITS)
  - SHARED CONTACT UPDATE
- ASSOCIATION
  - TRACK UPDATE/MISS PROCESSING (t^+ NODE)
  - TRACK SPAWN PROCESSING (t^- NODE)
  - CONTACT UPDATE/NEW TRACK/FALSE ALARM PROCESSING (C NODE)
  - SCENE PROCESSING (S NODE)

K9-7246/016

INFLUENCE DIAGRAM OPERATIONS APPLIED TO MIDCOURSE TRACKING

This chart summarizes the agenda for the remaining part of the presentation. It shows four major state estimation and four major association functions to be described in terms of Influence Diagram operations.

DISCRETE - TIME FILTERING

MATHEMATICAL MODEL

DYNAMIC PROCESS: $x(k+1) = \Phi(k)\,x(k) + \Gamma(k)\,w(k)$ $\quad k = 0, \ldots, N.$

MEASUREMENT PROCESS: $z(k) = H(k)\,x(k) + v(k)$ $\quad k = 0, \ldots, N.$

PROBABILISTIC STRUCTURE:

$E[x(0)] = \mu_0.$
$COV[x(0)] = P_0.$
$E[w(k)] = 0$ FOR $k = 0, \ldots, N.$
$COV[w(j), w(k)] = \delta_{jk} Q_k$ FOR $j = 0, \ldots, N$ AND $k = 0, \ldots, N.$
Q_k ARE DIAGONAL FOR $k = 0, \ldots, N.$
$COV[x(0), w(0)] = 0.$
$E[v(k)] = 0$ FOR $k = 0, \ldots, N.$
$COV[v(j), v(k)] = \delta_{jk} R_k$ FOR $j = 0, \ldots, N$ AND $k = 0, \ldots N.$
R_k ARE DIAGONAL FOR $k = 0, \ldots, N.$
$COV[w(j), v(k)] = 0$ FOR $j = 1, \ldots, N$ AND $k = 0, \ldots, N.$
$COV[x(0), v(k)] = 0$ FOR $0 = 1, \ldots, N.$

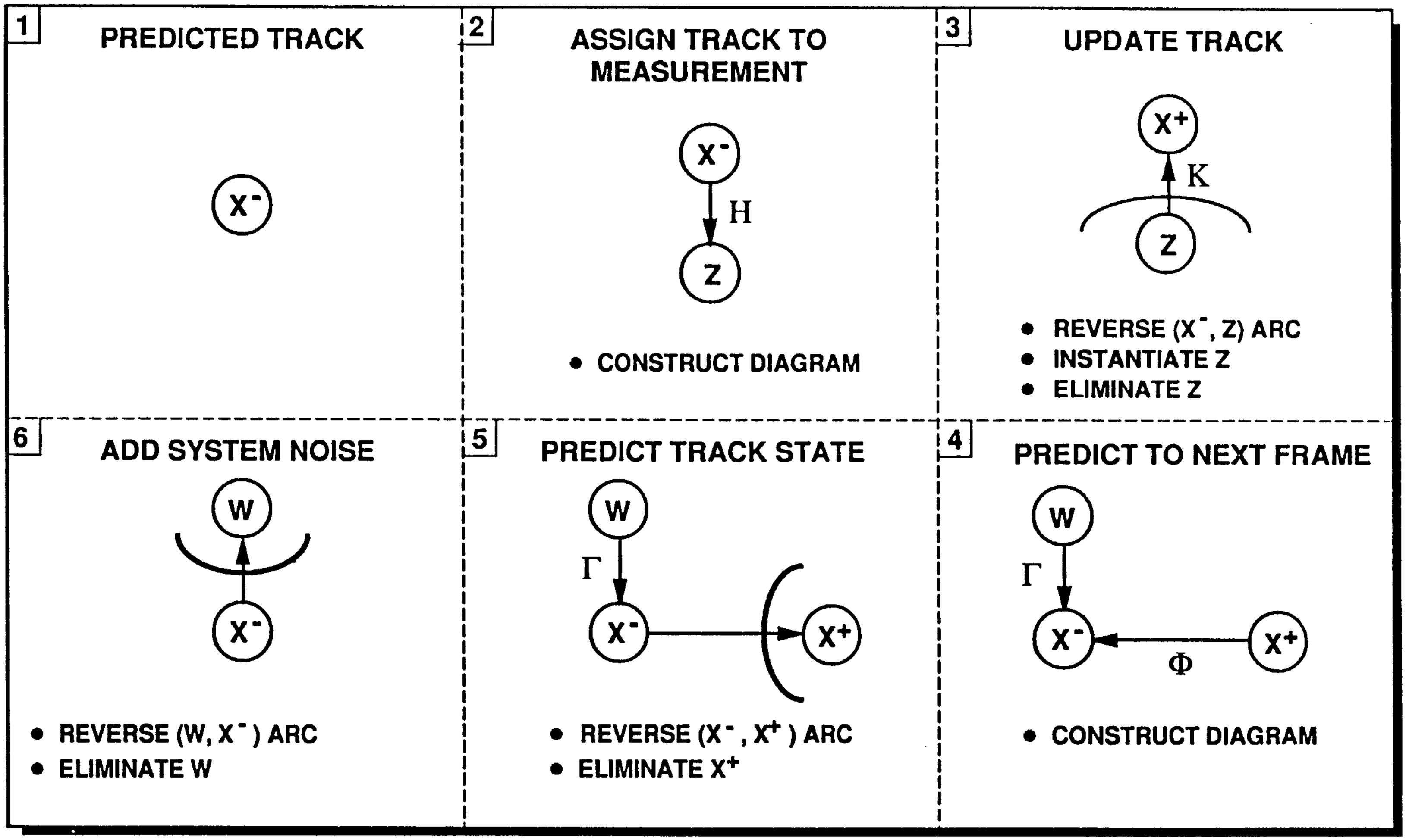
DIMENSIONS OF VECTORS: $x(k) \in R^n$, $w(k) \in R^r$, $z(k) \in R^P$, AND $v(k) \in R^P$.

K9-7246/009

DISCRETE - TIME FILTERING

This chart presents the assumptions made in the discrete-time Kalman filtering model.

KALMAN FILTER PROCESSING CYCLE



K9-7246/005

KALMAN FILTER PROCESSING CYCLE

This chart shows the cycle of Influence Diagram operations used to carry out the Kalman filter processing cycle.

[1] The state vector is predicted to the next measurement update.

[2] A measurement is assigned to the state vector. An arc from X- to Z in constructed and the measurement matrix, H, is placed on the arc.

[3] The track is updated. First the arc is reversed which updates the covariance matrx for the state vector. Next the Z node is instantiated which updates the mean of the state vector.

[4] The track is set up for prediction to the next frame. First, a construction phase is carried out in which an arc is created between the updated state vector, X+, and the (to be) predicted state, X-. Also, an arc from the system noise vector, W, to the predicted state is also constructed. The propagation matrix is placed on the X+ to X- arc, and the Gamma matrix is placed on the W to X- arc.

[5] The track is predicted. The X+ to X- arc is reversed which calculates the predicted covariance matrx. (Note that the predicted state is calculated outside of the Influence Diagram for the case of the extended Kalman filter.)

[6] The system noise is incorporated. The W to X- arc is reversed which has the effect of adding the Q matrix to the predicted covariance.

DISCRETE - TIME FILTERING

WEIGHTED OPERATION COUNTS FOR PROCESSING A VECTOR OF p MEASUREMENTS

ALGORITHM	WEIGHTED OPERATION COUNTS
INFLUENCE DIAGRAM	$(3.6n^2 + 12.3n)\,p$
CONVENTIONAL KALMAN	$(3.6n^2 + 7.8n + 4.5)\,p$
U-D COVARIANCE	$(3.6n^2 + 15.7n)\,p$
SQUARE ROOT COVARIANCE	$(4.3n^2 + 40.9n)\,p$
POTTER SQUARE ROOT	$(7.2n^2 + 8.6n + 30.4)\,p$
KALMAN STABILIZED	$(10.1n^2 + 16n + 4.5)\,p$
SRIF, R TRIANGULAR	$(2.4n^2 + 6.2n)\,p + 4.3n^2 + 37.1n$
NORMAL EQUATION	$(1.2n^2 + 5.0n)\,p + 0.4n^3 + 3.1n^2 + 30.3n$
SRIF, R GENERAL	$2.4n^2 p + 1.6n^3 + 2.6n^2 + 31n$

OPERATION	WEIGHT
$+$	1
X	1.4
$\div$	4.5
$\sqrt{}$	21.4

K9-7246/011

DISCRETE-TIME FILTERING

This chart shows the weighted operation counts for the update phase of the Kalman filter processing cycle. It can be seen that the Inluence Diagram implementation compares favorably with the conventional Kalman implementation which is the most efficient. The Influence Diagram implementation requires less throughput than the other versions shown.

One other advantage of the Influence Diagram implementation is that it guarantees a positive semidefinite covariance matrix since the variance terms are calculated by summing positive quatities. Subtractions, which can cause numerical instabilities, are not required.

DISCRETE - TIME FILTERING

WEIGHTED OPERATION COUNTS FOR TIME UPDATE

ALGORITHM	WEIGHTED OPERATION COUNTS
INFLUENCE DIAGRAM	$2.8n^3 + 3.95n^2 - 11.55n + 10 + (6n^2 + 2.7n - 4.9)\,r + (2.4n - 2.4)\,r^2$
CONVENTIONAL KALMAN	$3.6n^3 + 4.1n^2 + 0.5n + (1.2n^2 + 2.6n)\,r$
U-D COVARIANCE	$3.6n^3 + 4n^2 + 3.1n - 4.5 + (2.4n^2 + 4.2n - 2.8)\,r$
SQUARE ROOT COVARIANCE	$4n^3 + 4.8n^2 + 26.7n + (2.4n^2 + 2.4n)\,r$

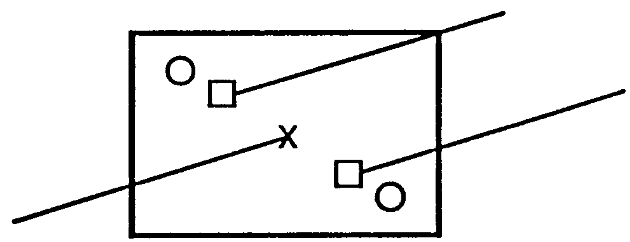
K9-7246/010

DISCRETE-TIME FILTERING

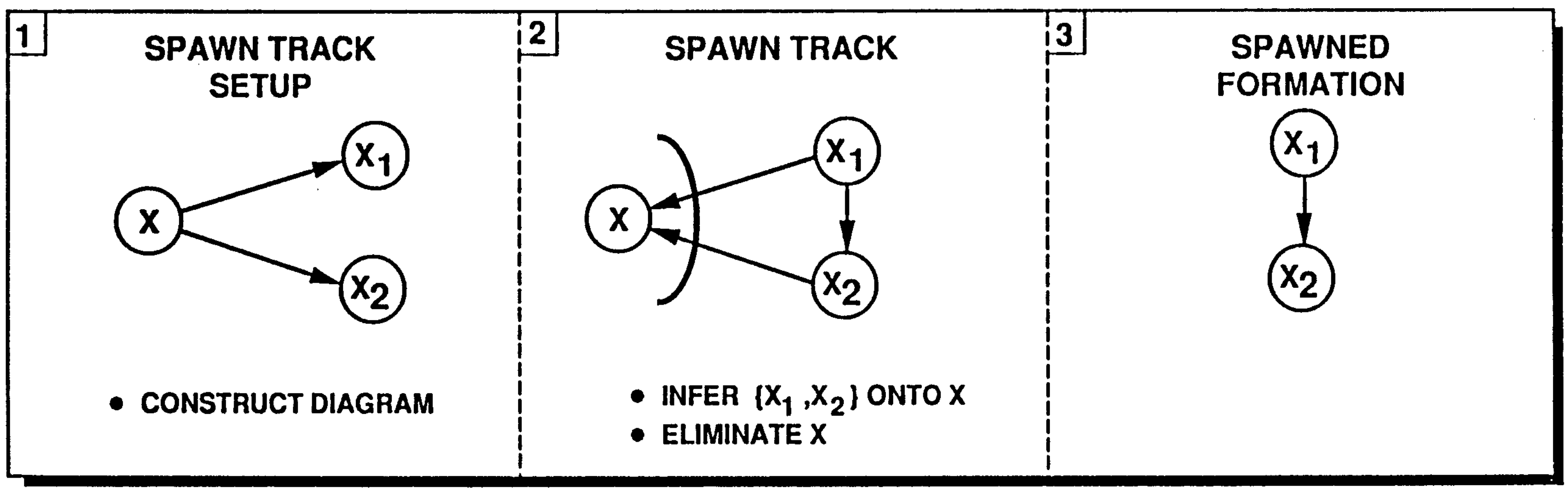
This chart shows the weighted operation counts for the time update portion of the Kalman filter. Again, the Influence Diagram implementation performs well against the implementations shown.

TRACK SPAWNING ('AND' SPLITS)

- SPLIT TRACK X INTO X_1 AND X_2
- X_1 & X_2 BECOME A TWO-TRACK FORMATION



- □ ESTIMATED STATE
- X PREDICTED STATE
- ○ CONTACT



K9-7246/006

TRACK SPAWNING ('AND' SPLITS)

This chart shows the Influence Diagram operations involved in performing a splitting of a track into two tracks. Suppose two contacts fall into the correlation gate for the track and it is determined that the track has split into two tracks.

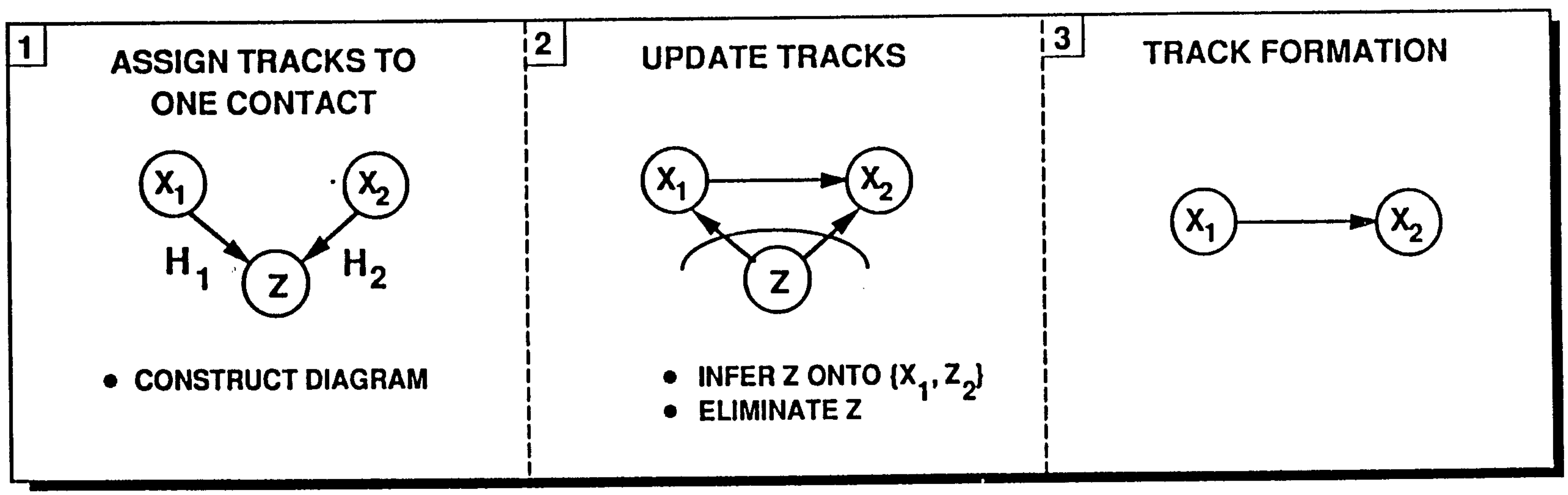
[1] The Influence Diagram for a Track Spawn is constructed. Two new state vectors are created and an arc from the previous state vector, X, to the two new states, X1 and X2, are constructed. Next, the appropriate arc strengths are placed on the arcs. Likewise, the data in the X1 and X2 vertices are set. The data includes the unconditional means, conditional variances and internal arc strengths.

[2] The arcs between X1 and X2 and X are reversed. As a result, and arc from X1 and X2 is created. Thus a two track formation is created. The strength of the arc between the tracks depends upon the conditional variances set in X1 and X2 and the arc strengths from X to X1 and X2. The term, 'formation', is used whenever an influence arc exists between the tracks.

[3] The X node is deleted leaving the spawned formation.

SHARED CONTACT UPDATE

- TWO TRACKS SHARE THE SAME CONTACT. BOTH TRACKS ARE UPDATED WITH SAME CONTACT



K9-7246/007

SHARED CONTACT UPDATE

This chart shows the Influence Diagram operations involved in updating two tracks with the same contact. This situation occurs when the sensor has less resolution than the track file or the tracks are crossing from the sensor's perspective.

[1] The contact is assigned to both tracks. The Influence Diagram is constructed with arcs from the tracks, X1 and X2, to the contact, Z. The measurement matrices are placed on the arcs and the appropriate data is set in the Z node.

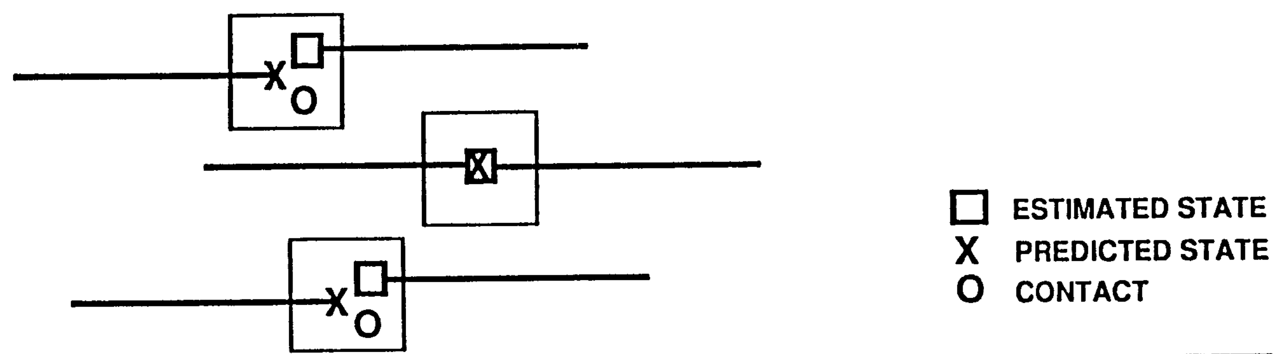
[2] Inference of the Z node onto the X nodes is carried out and the Z node instantiated and then eliminated. This stage updates the covariances and state vectors of the tracks.

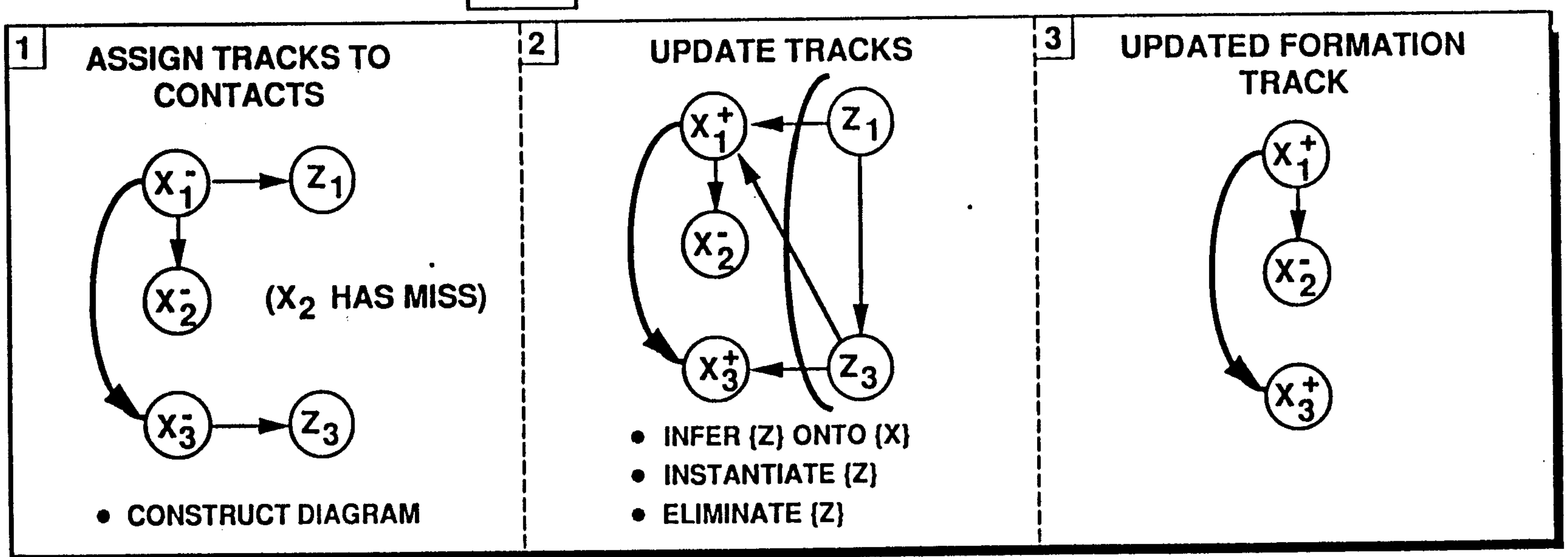
[3] A two track formation is created.

It should be noted that one of the advantages of the Influence Diagram implementation is that all relevent probabilistic influences are maintained automatically as part of the Influence Diagram operations.

FORMATION TRACK UPDATE

- UPDATE FORMATION TRACK WITH A SET OF CONTACTS





K9-7246/012

FORMATION TRACK UPDATE

This chart shows the Influence Diagram operations involved in updating a formation track.

The example shows a 3 track formation in which two of the tracks have an update and the third track has a miss.

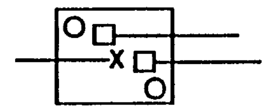
[1] The contacts are assigned to the tracks and the Influence Diagram is constructed.

[2] The tracks are updated. Inference of the measurements onto the state vectors is carried out; the measurement nodes are instantiated and then eliminated.

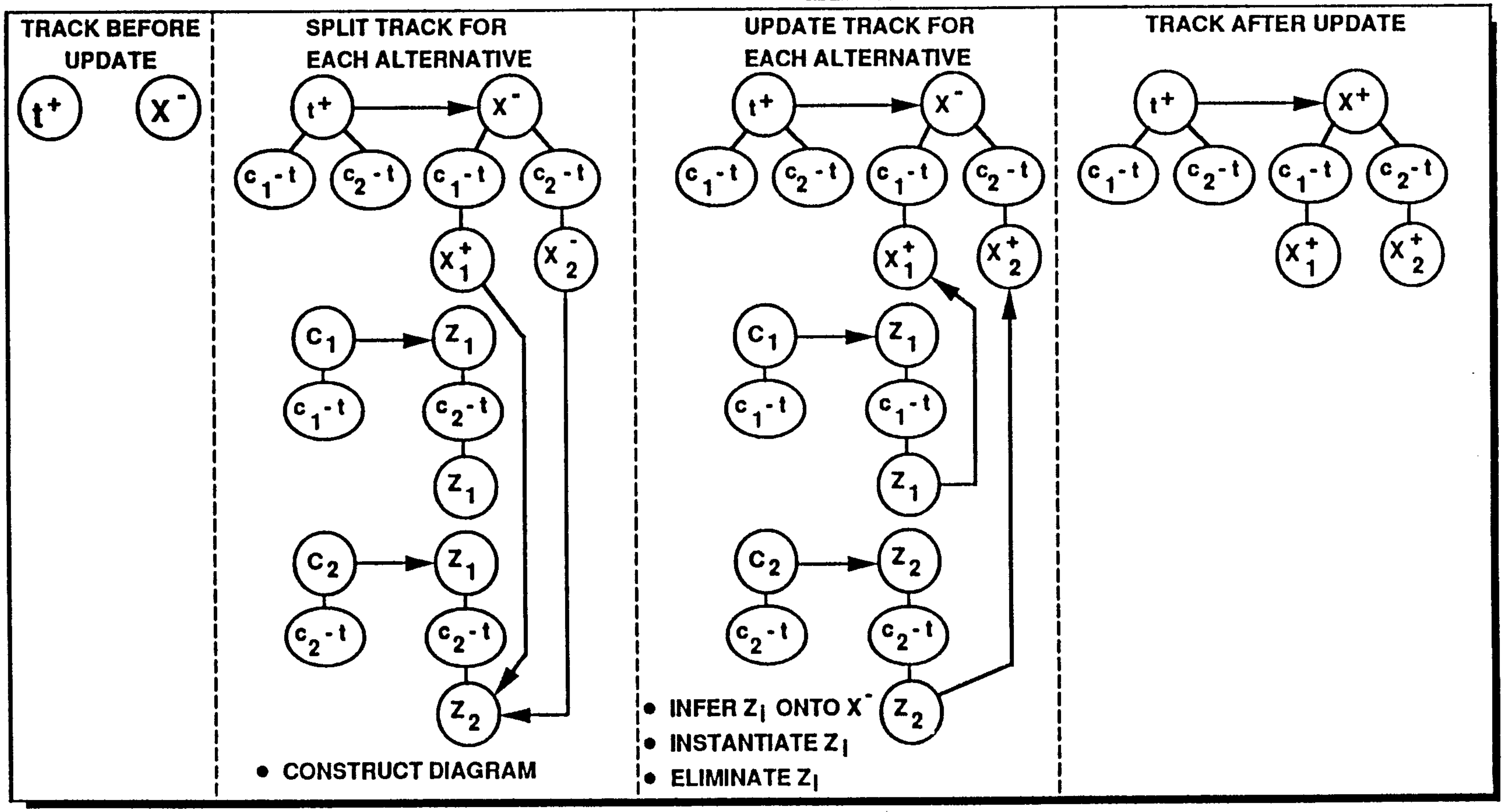
[3] The formation track is updated. Note that the track with the miss maintains the propagated covariance and state vector.

t^+ TRACK SPLITTING (' OR ' SPLITS)

- TRACK HAS MULTIPLE CONTACTS IN THE GATE



X PREDICTED STATE
□ UPDATED STATE
O CONTACT



K9-7246/017

t+ TRACK SPLITTING ('OR' SPLITS)

This chart shows the Influence Diagram operations involved in splitting a track into alternative tracks due to multiple contacts in the gate. In this example, two contacts fall into the gate.

[1] Before the update processing, a single state vector, X-, and an unelaborated update assignment node, t+, exist.

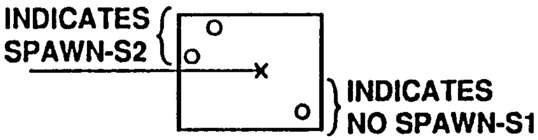
[2] The contact assignments are added as outcomes to the t+ node. An arc is added from the t+ node to the X- node which causes the outcomes to flow to the X- node, thereby creating two x node versions. (An error exists in the chart. X1+ should be X1- and should have an arc to Z1.) The update assignments are also placed under the C nodes and arcs created from the C nodes to the Z nodes. Finally, in order to start the update process, a continuous arc from X1- to Z1, and X2- to Z2 are created and the measurement matrices placed on the arcs.

[3] The arcs are reversed; the Z nodes instantiated and then eliminated.

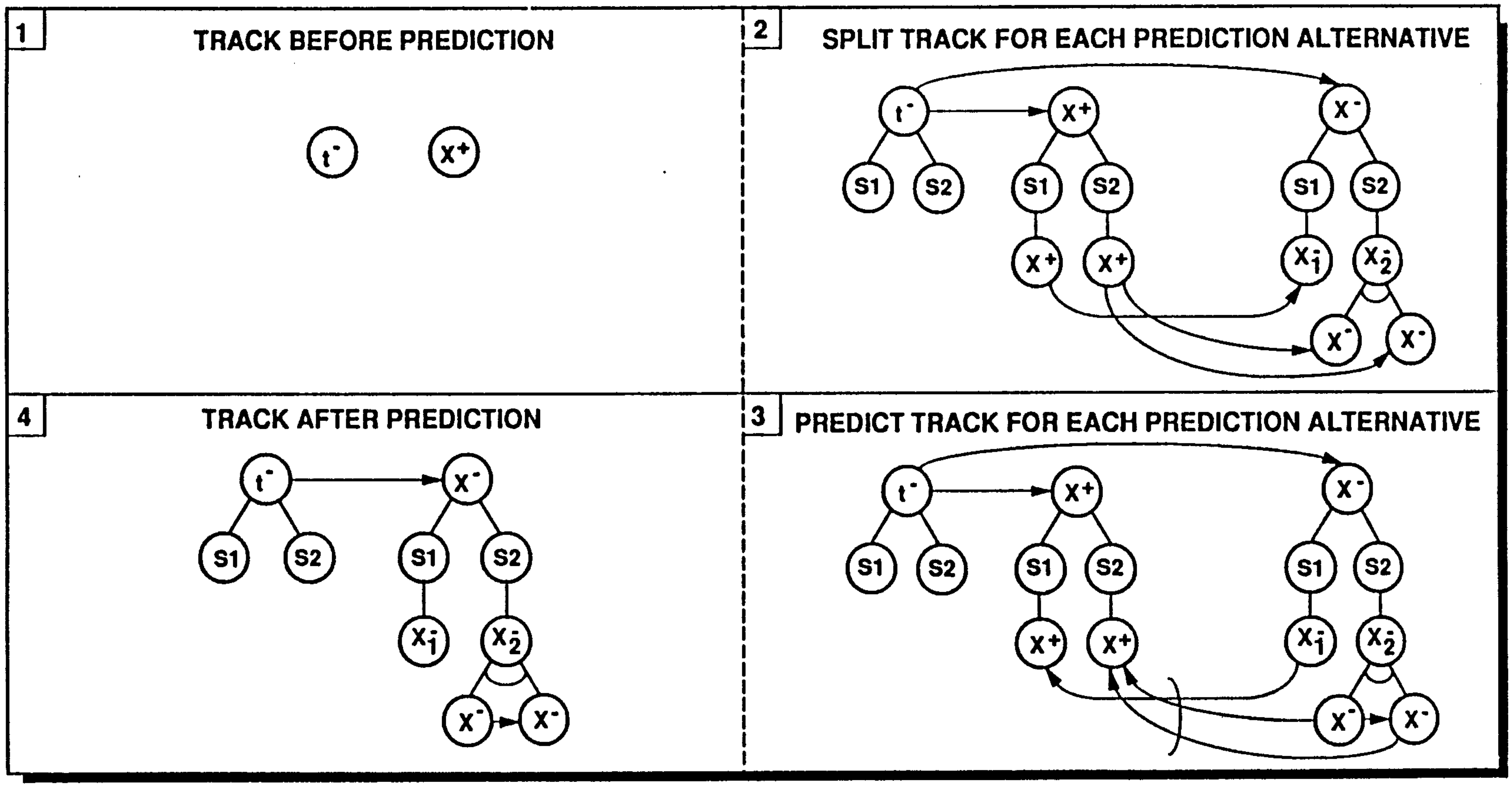
[4] The track after the update shows two updated versions of the track state vector.

t^- NODE PROCESSING

- TRACK HAS MULTIPLE SPAWN ALTERNATIVES



X = PREDICTED STATE
O = CONTACT



K9-7246/018

t- NODE PROCESSING

This chart shows the Influence Diagram operations involved with track prediction processing. This process considers alternative spawn hypotheses in which a track splits into more than one track thereby creating a formation.

[1] Before the prediction processing, an updated state vector exists and an unelaborated t- node.

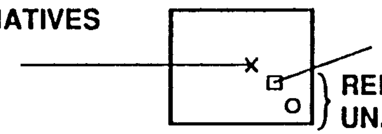
[2] It is determined that the track either spawns a 2 track formation or remains a singleton track. As a result, an S2 assignment and an S1 assignment are added as outcomes to the t- node. An arc is added from t- to the X+ node and the outcomes flow to the X+ node creating two alternatives. Likewise, a predicted state vector, X-, iscreated and an arc from t- to X- is created and the outcomes flow to create two versions for X-. The S2 version creates 2 state vectors. Finally, arcs are added from the X+ versions to the X- versions and the data set on the arcs and in the nodes.

[3] The arcs are reversed and the predicted state vectors are calculated. The X+ node can then be deleted.

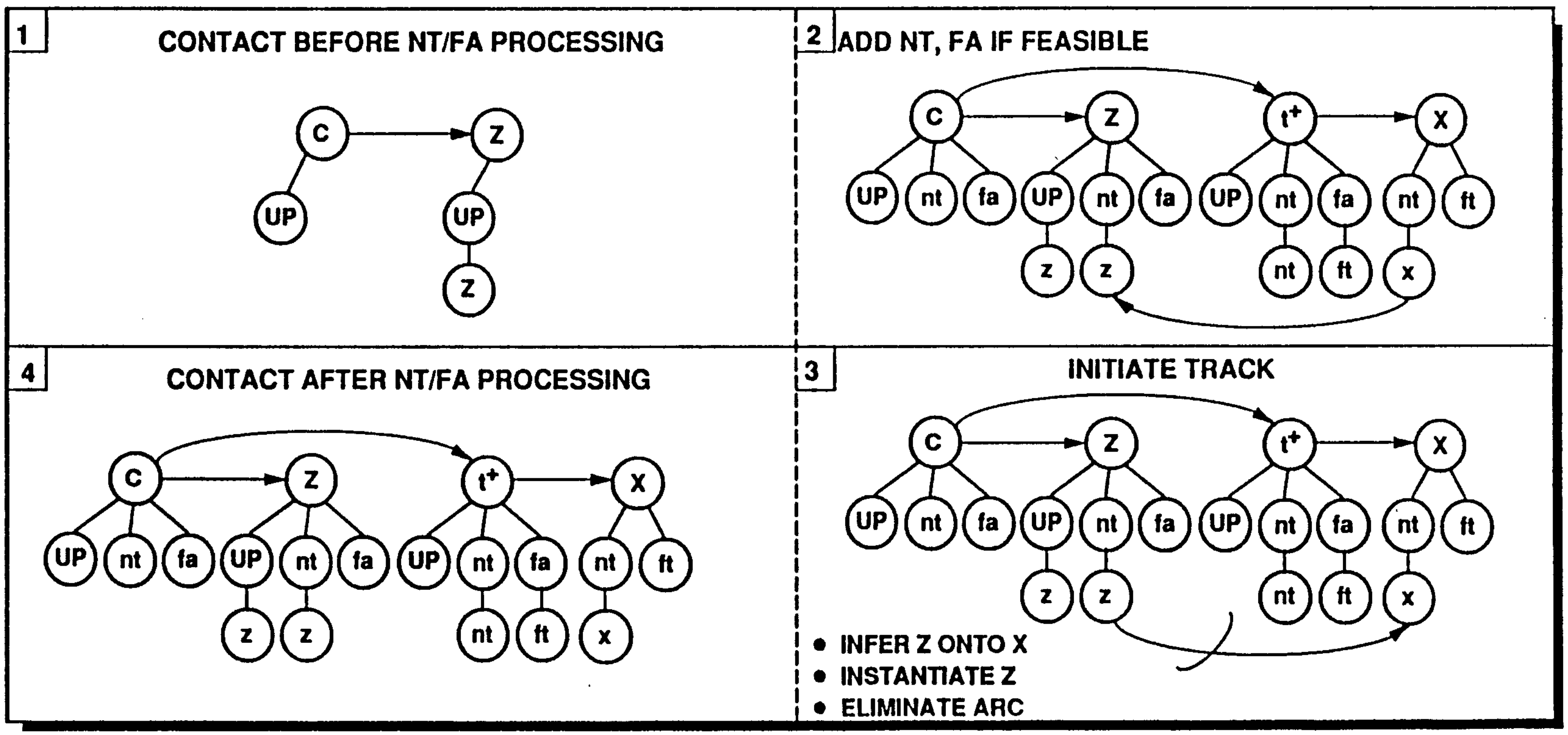
[4] After prediction processing, a predicted 2-track version and a singleton track version exist.

C NODE PROCESSING

- CONTACT FALLS IN GATE BUT IS NOT A GOOD FIT TO TRACK SO NEW TRACK AND FALSE ALARM ARE FEASIBLE ALTERNATIVES



□ = UPDATED STATE
X = PREDICTED STATE
O = CONTACT



NOTE: UP = ASSIGNMENT TO A TRACK
NT = NEW TRACK
FA = FALSE ALARM

K9-7246/019

C NODE PROCESSING

This chart shows the Influence Diagram operations involved with processing a C node. In this example, the contact fell into a track gate but fell near the edge of the gate. Therefore, there is a reasonable likelihood that it may be a new track or false alarm.

[1] Before processing the C node, the update assignment was added at the time the t+ node was processed. (See the chart on t+ node processing.)

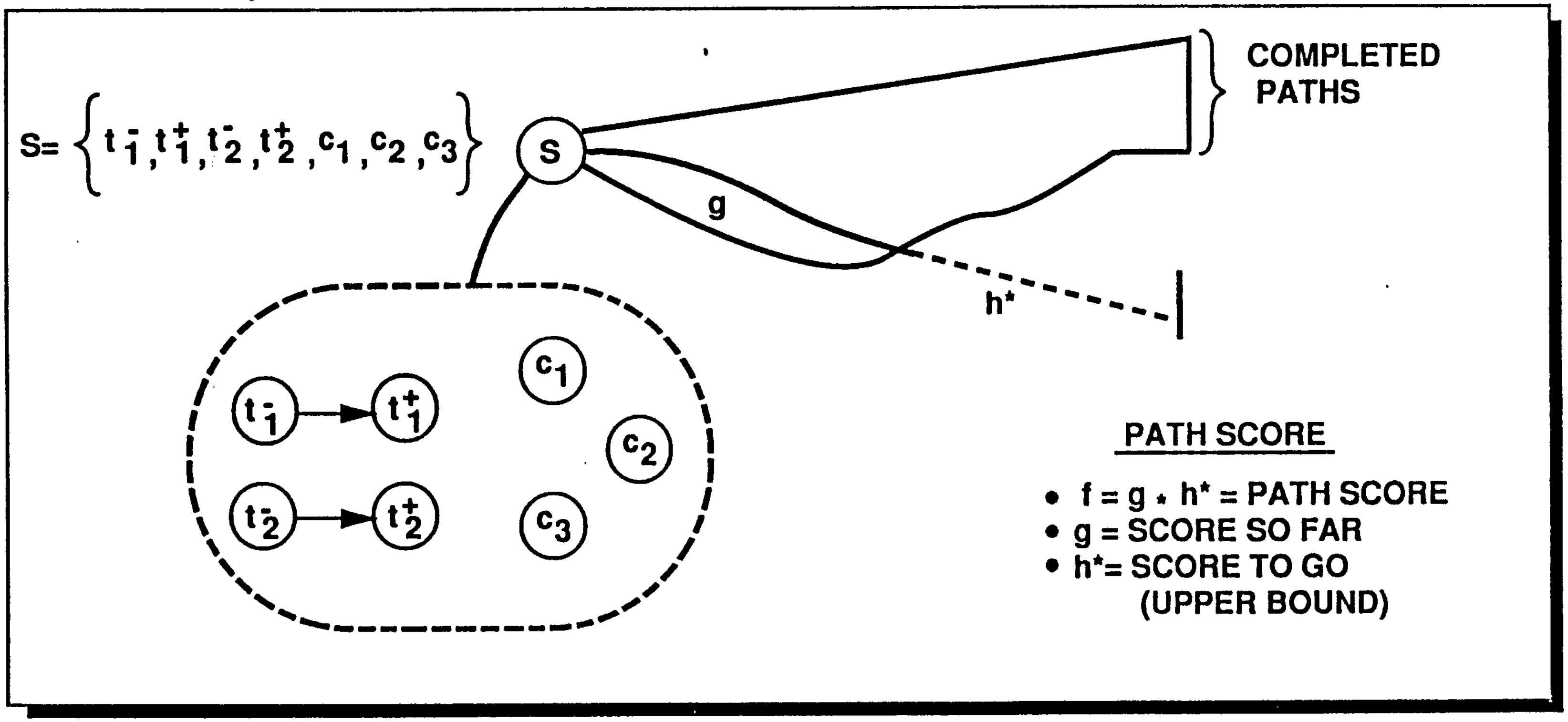
[2] The new track and false alarm outcomes are added and flow to the Z node creating new versions. A Z node state vector is not created for the false alarm version. For the new track version, a t+ node iscreated and an arc from the C node to the t+ node is created. Likewise, an X node is created and an arc from the t+ node to the X is created. Finally, an arc from the new track version for the X node to the new track version for the Z node is created and the data is set.

[3] The arc is reversed; the Z node instantiated and the arc then eliminated.

[4] After C node processing in this example, a new track is created.

S NODE PROCESSING

- S NODE TRAVERSAL USES A* TREE SEARCH TO FIND ALL JOINT OUTCOMES (PATHS) SUCH THAT $f_{path} \geq (1-\alpha) f_{best}$



$$g = \underbrace{\pi(\beta_{up_i} PD_i)}_{NUP} \; \underbrace{\pi(1-PD_i)}_{NMISS} \; \underbrace{(\beta_{FA})}_{NFA} \; \underbrace{(\beta_{NT})}_{NNT}$$

h*= UPPER BOUND ON SCORE FOR REMAINING TRACKS AND CONTACTS

K9-7246/027

S NODE PROCESSING

This chart summarizes the processing associated with the scene node. The scene node, S, is a vector discrete node the represents the joint outcomes of a set of t+, t- and C nodes. The S node processing attempts to generate all feasible joint outcomes by using a search strategy based on the A* tree search algorithm.

The A* algorithm tries to find the best path in a tree by calculating the score so far, g, for a path and an upper bound on score to go, h*. The scores are combined to create a score for each path, f, in the tree. The path with the highest score is then used for continuing the search for the best complete path.

The score so far, g, is a function of the number of updates and their likelihoods, misses, new tracks and new track density, and false alarms and false alarm density on the path so far.

The A* approach is used to find all complete paths, i.e. joint outcomes, that have scores within a certain distance to the best path.

In this example, the S node represents the a sete of seven random variables.

S NODE PROCESSING

SCENE PROCESSING (S)

LOOP
1. SELECT BEST INCOMPLETE PATH ABOVE THRESHOLD
2. IF NULL, EXIT
3. GET NEXT NODE
4. ELABORATE NODE
5. EXTEND PATH

END LOOP

UPDATE TRACK PROCESSING (t^+)

1. GET FEASIBLE OUTCOMES
2. PERFORM ACTION FOR EACH OUTCOME:

UPDATE	MISS	FALSE TRACK
• UPDATE STATE • PRUNE ACTION	• UPDATE DATA • PRUNE ACTION	• NO ACTION

SPAWN TRACK PROCESSING (t^-)

1. GET FEASIBLE OUTCOMES
2. PERFORM ACTION FOR EACH OUTCOME:

NO SPAWN	SPAWN INTO K	FALSE TRACK
• NO ACTION	• SPAWN ACTION	• NO ACTION

CONTACT PROCESSING (C)

1. GET FEASIBLE OUTCOMES
2. PERFORM ACTION FOR EACH OUTCOME:

UPDATE	NEW TRACK	FALSE ALARM
• MERGE ACTION	• INITIATE TRACK	• NO ACTION

K9-7246/026

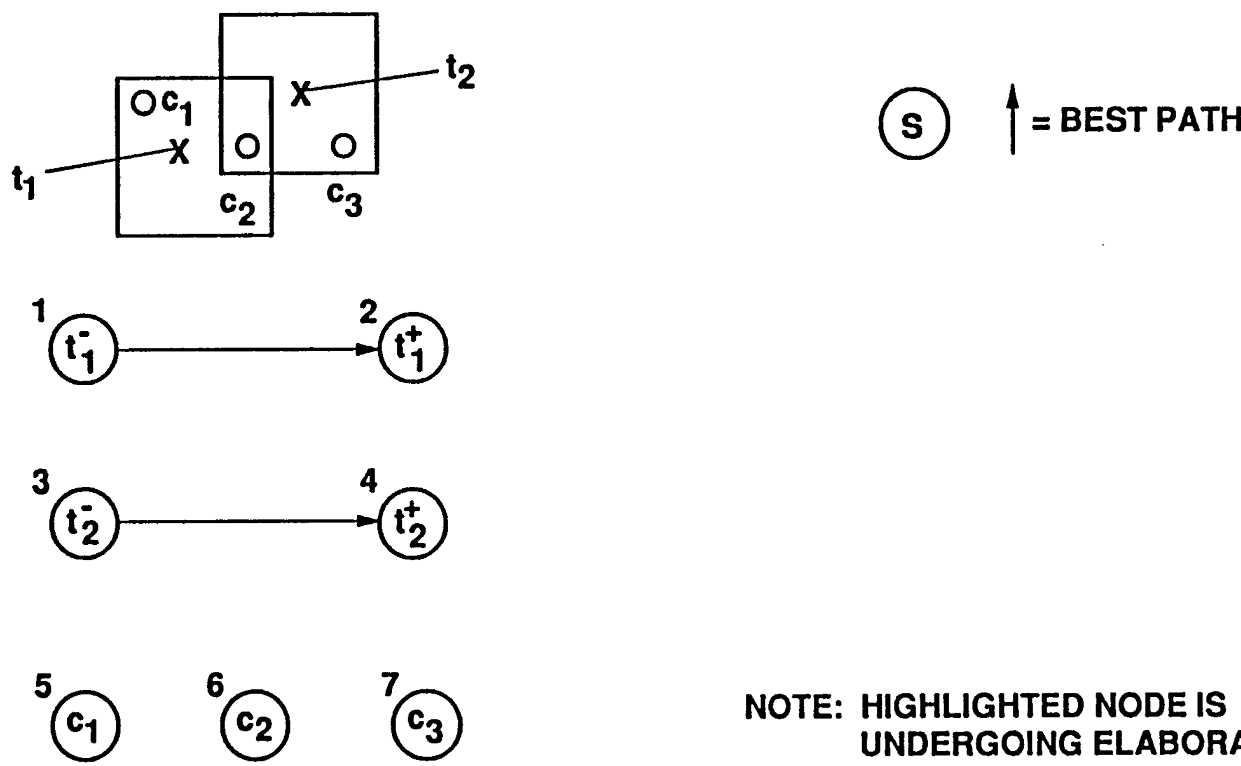
S NODE PROCESSING

This chart outlines the process flow in elaborating the S node. A search loop is executed which carries out the A* search algorithm. Once a node is selected for elaboration, then the node is elaborated. Three elaboration routines exist: Elaborate t+ node which performs update track processing, elaborate t- node which carries out track spawn processing, and elaborate C node which performs contact processing.

The basic algorithm architecture decomposes into a global search which controls a localized elaboration process.

For each node and for each outcome added to the node an action is performed that is specific to the outcome added.

SCENE PROCESSING EXAMPLE – 0



NOTE: HIGHLIGHTED NODE IS UNDERGOING ELABORATION

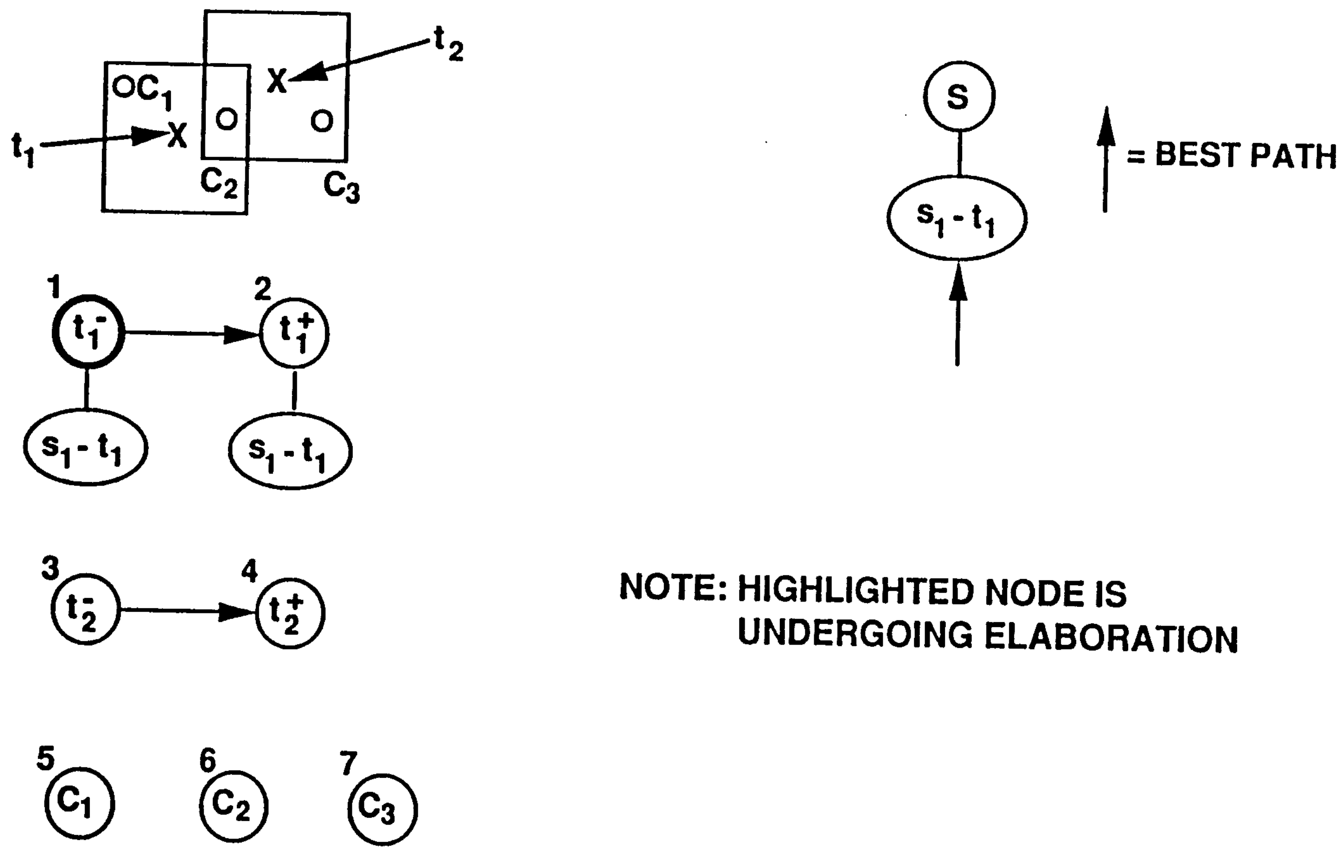
K9-7246/048

SCENE PROCESSING EXAMPLE - 0

In the next 9 charts, the steps in Scene processing are illustrated. The example chosen consists of two tracks with overlapping gates. Three contacts fall in the gates with one contact in the overlap region.

The S node consists of 7 nodes, and the S node and all 7 nodes are shown unelaborated. For the following charts, the best path is illustrated with the up arrow and the node undergoing elaboration is highlighted. The order in which the nodes are visited are numbered.

SCENE PROCESSING EXAMPLE – 1



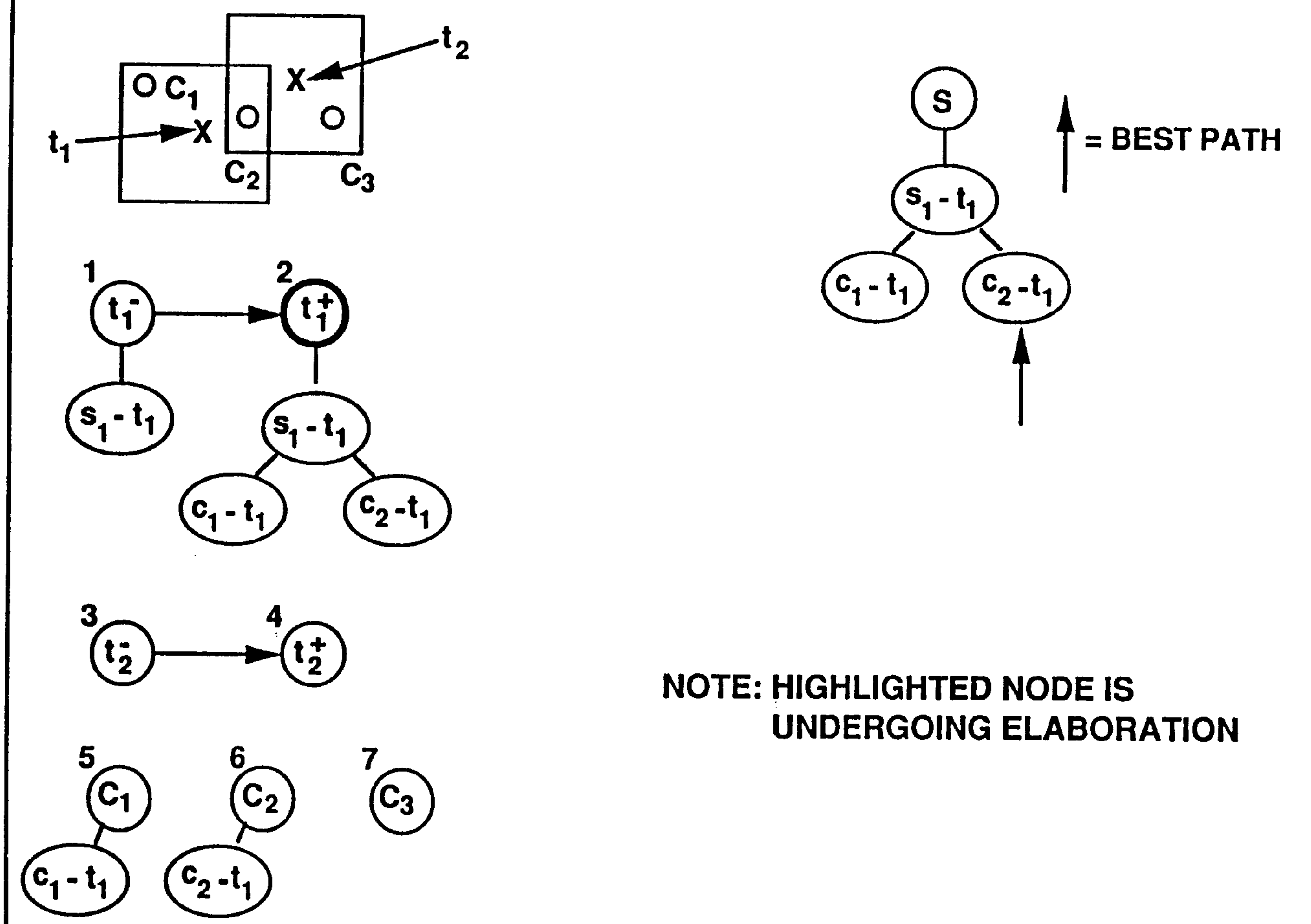
NOTE: HIGHLIGHTED NODE IS UNDERGOING ELABORATION

K9-7246/049

SCENE PROCESSING EXAMPLE - 1 to 9

This sequence of charts show the step by step elaboration of the Scene node and the nodes within the scene. At each stage, the best path in the S node joint outcome tree is determined. The next node to be elaborated is selected and the node is elaborated. During the elaboration process, the continuous nodes are managed. When all paths are generated that are within a certain tolerance of the best path are generated the S node elaboration process halts.

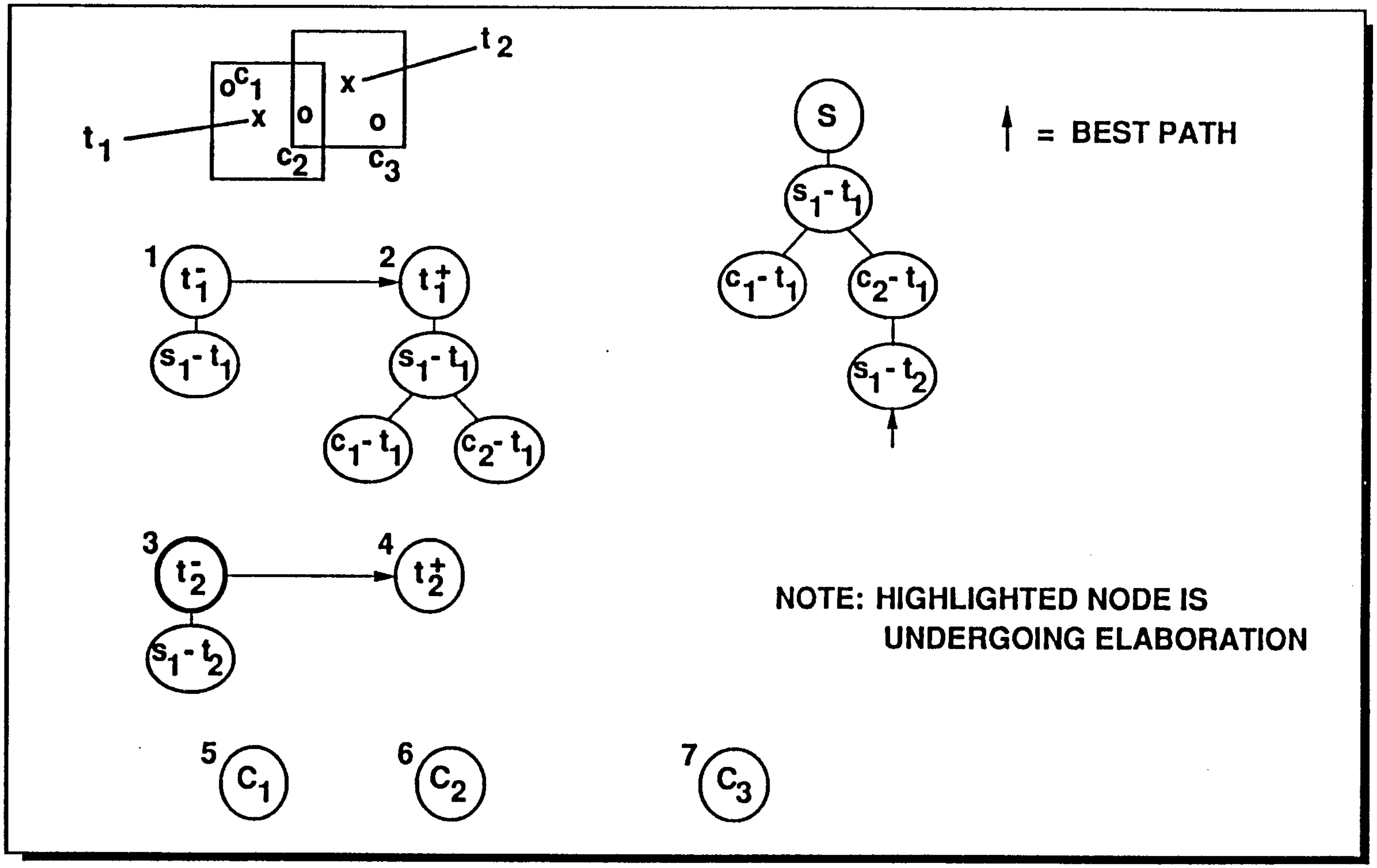
SCENE PROCESSING EXAMPLE – 2



NOTE: HIGHLIGHTED NODE IS UNDERGOING ELABORATION

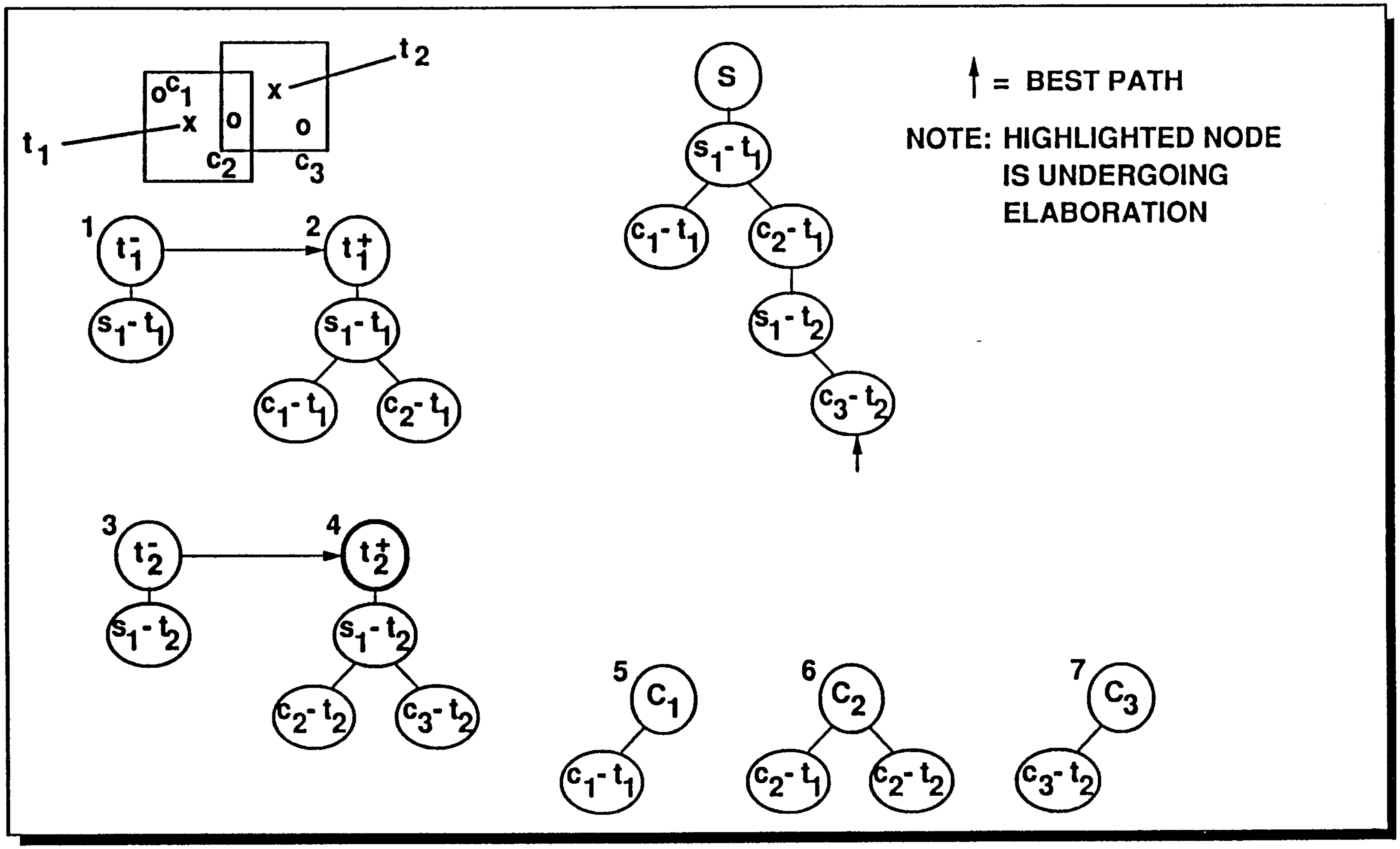
K9-7246/050

SCENE PROCESSING EXAMPLE – 3



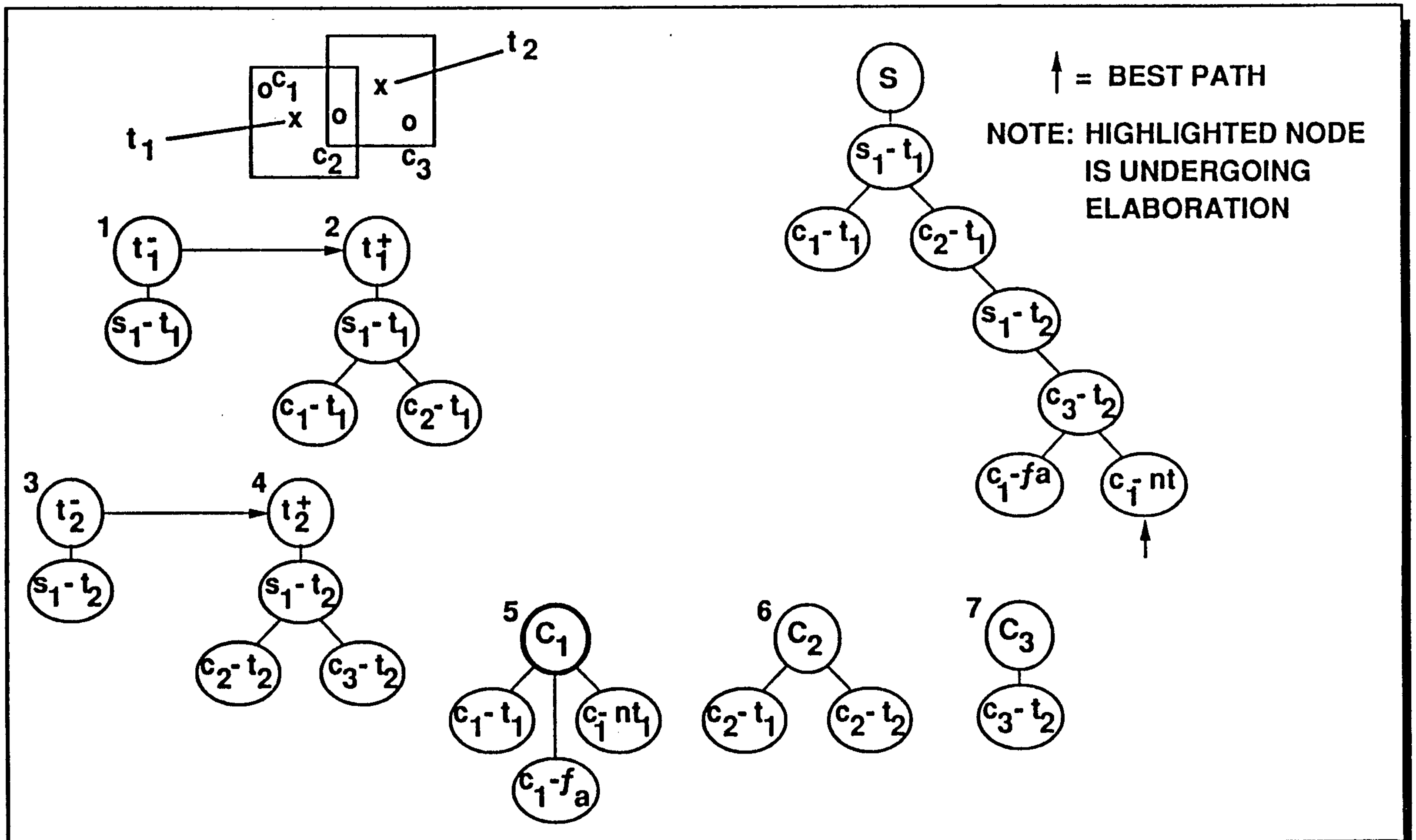
K9-7246/051

SCENE PROCESSING EXAMPLE – 4



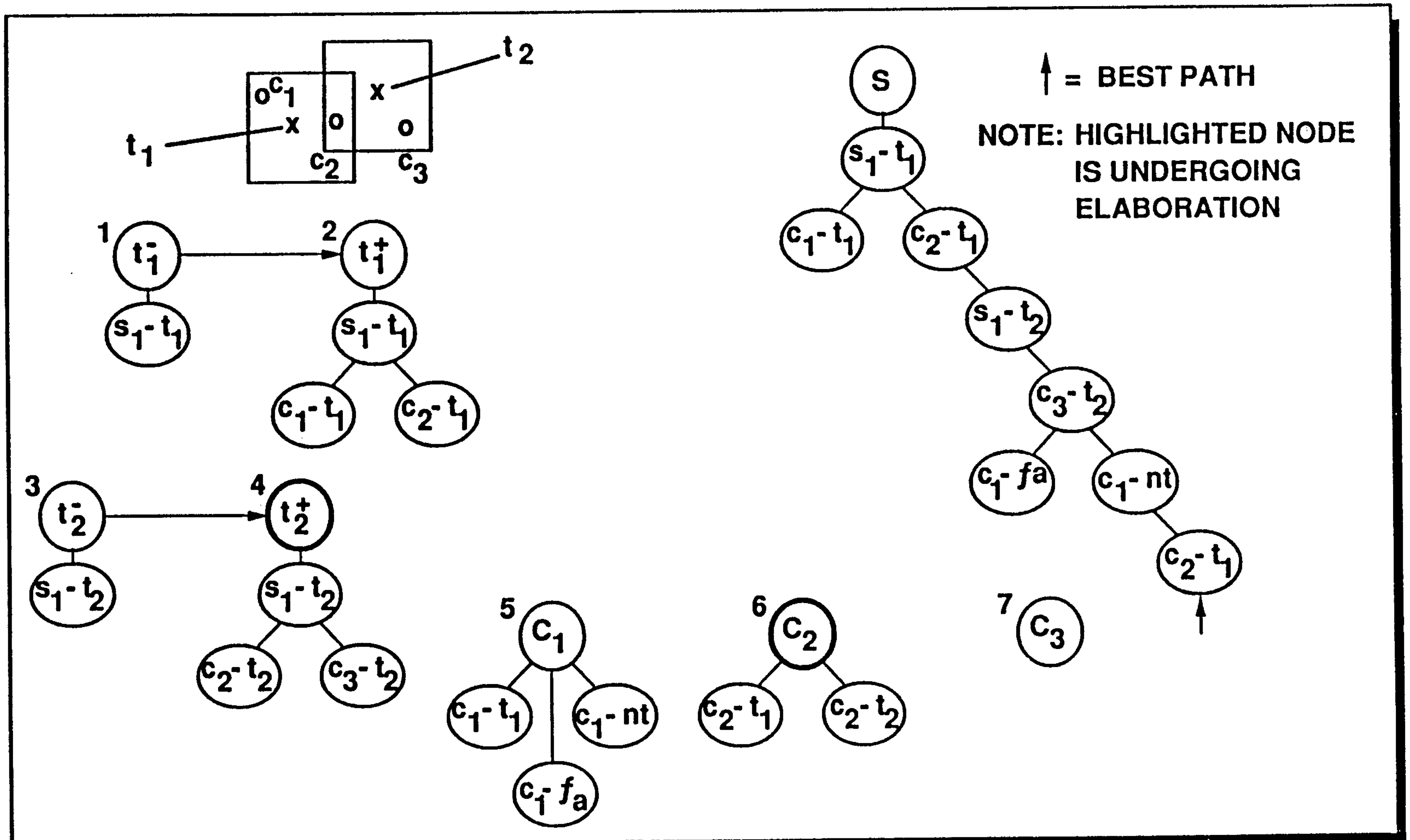
K9-7246/052

SCENE PROCESSING EXAMPLE – 5



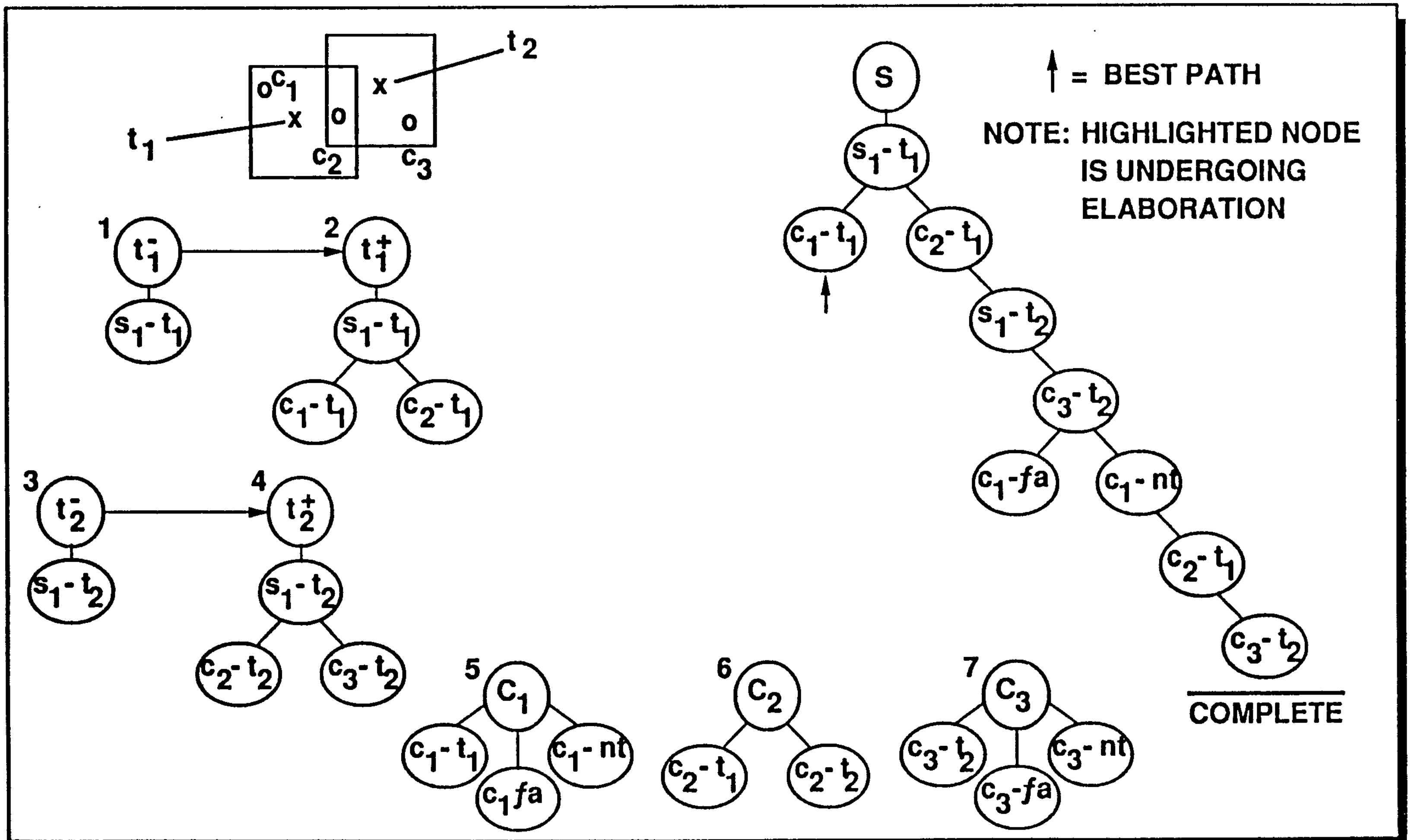
K9-7246/053

SCENE PROCESSING EXAMPLE – 6



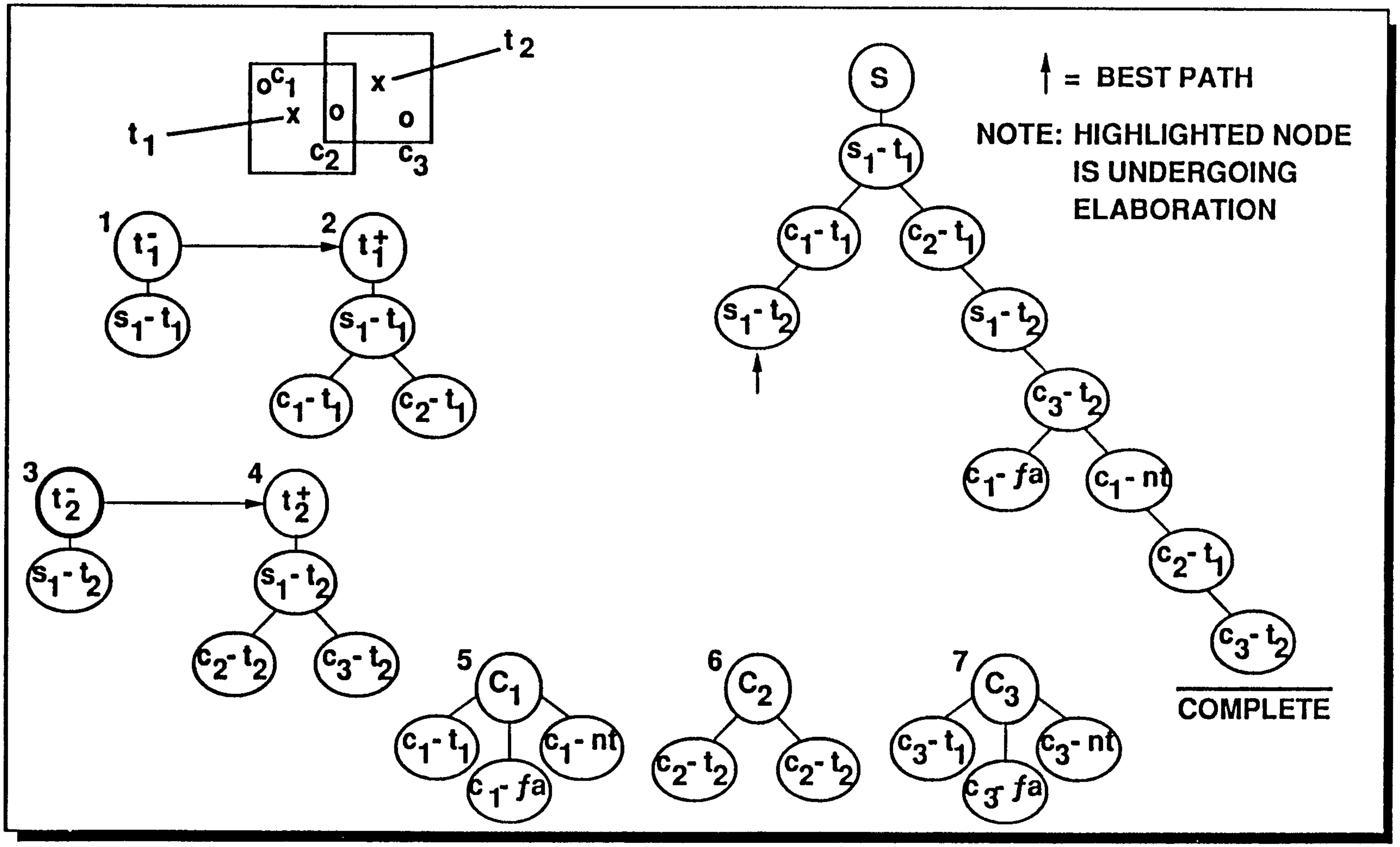
K9-7246/054

SCENE PROCESSING EXAMPLE – 7



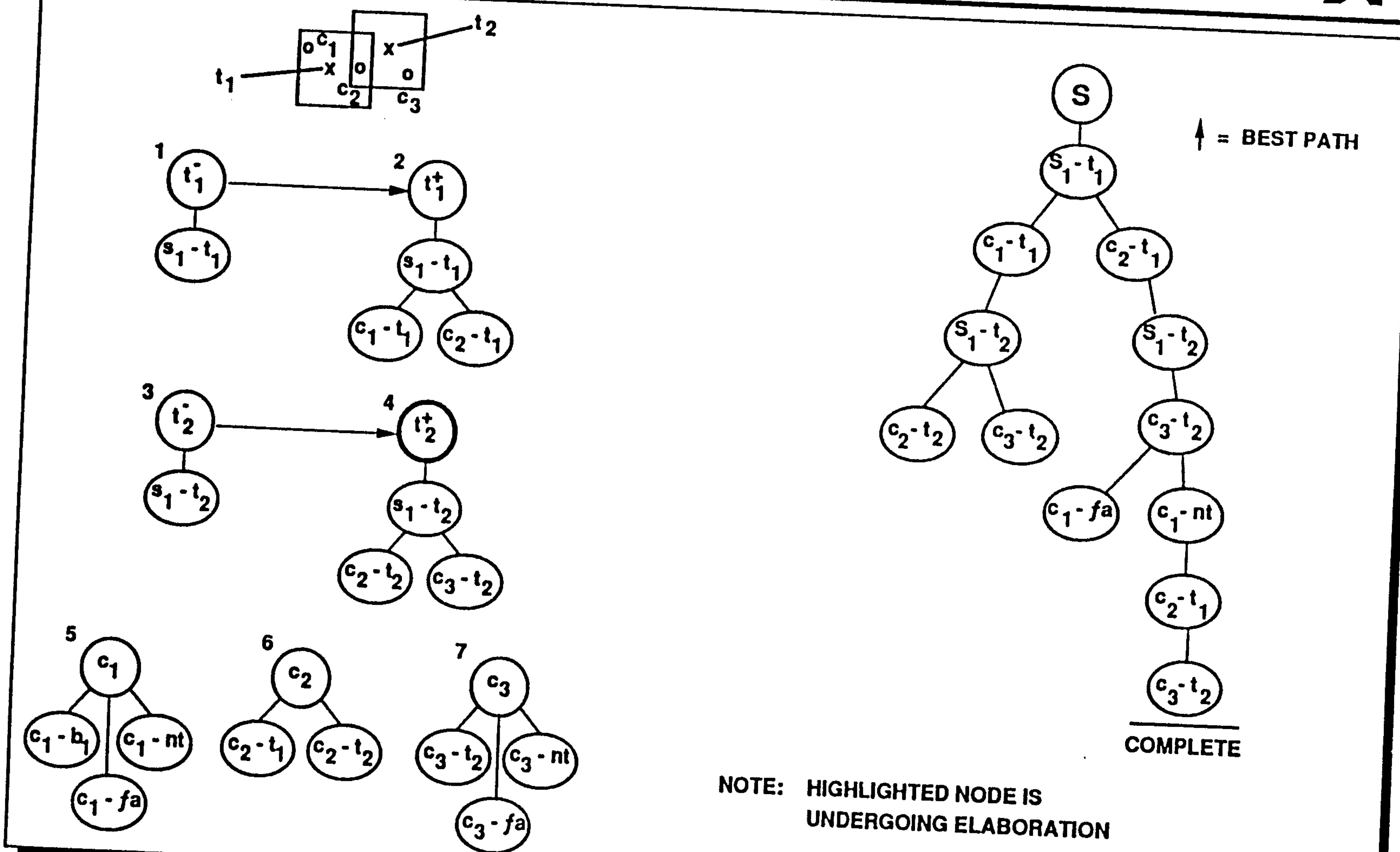
K9-7246/056

SCENE PROCESSING EXAMPLE – 8



K9-7246/055

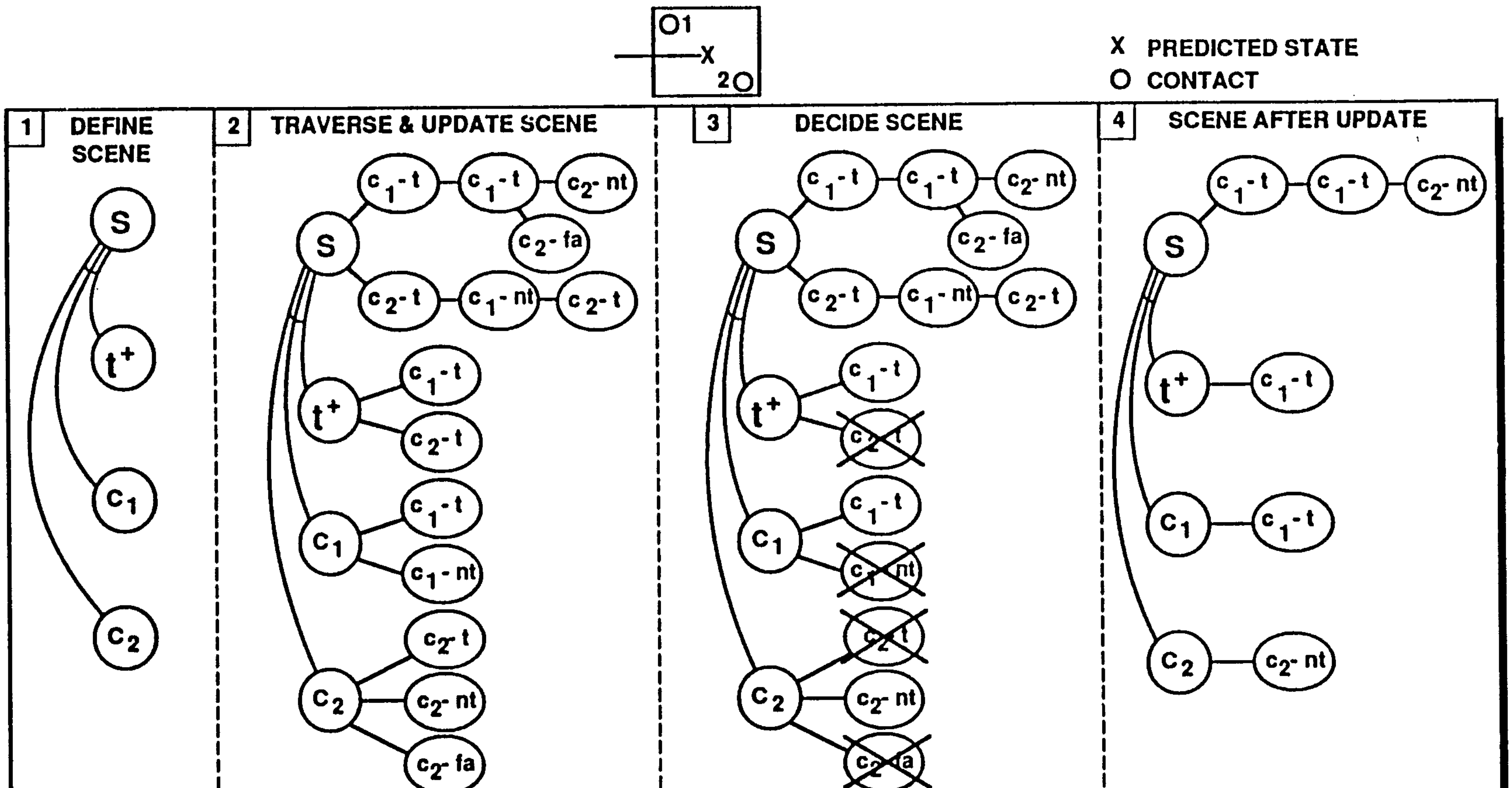
SCENE PROCESSING EXAMPLE – 9



NOTE: HIGHLIGHTED NODE IS UNDERGOING ELABORATION

K9-7246/058

SCENE PROCESSING EXAMPLE



- USE A* TREE SEARCH TO DEVELOP ALL SIGNIFICANT JOINT OUTCOMES FOR S

- PROJECT S INTO DECISION NODEL(S)
- PERFORM BAYES' TEST

K9-7246/028

SCENE PROCESSING EXAMPLE

This chart shows the complete processing for the Scene node for the example of two contacts in the track gate.

[1] The Scene node is defined as consisting of the update node for the track, t+, and the two contact nodes, C1 and C2.

[2] The traversal algorithm based on A* search is carried out to elaborate the S node and the t+ , C1 and C2 nodes.

[3] n The likelihoods calculated for the joint outcomes in the Scene node are projected onto the t+, C1 and C2 nodes. A Bayes' decision test is performed to select an outcome and prune the other outcomes. Pruning outcomes also removes versions on the continuous side of the Influence Diagram.

[4] After the scene decision process, one outcome remains in this example: C1 updates the track and C2 begins a new track.

SUMMARY

- SHOWED HOW THE INFLUENCE DIAGRAM CAN BE USED TO:
  - REPRESENT PROBABILISTIC INFORMATION GENERATED IN MIDCOURSE TRACKING
  - CARRY OUT:
    - STATE ESTIMATION (UPDATE, PREDICTION, SPAWNING)
    - DATA ASSOCIATION (HYPOTHESIS GENERATION, SCORING AND SELECTION)
    - TRACK PROMOTION
- CURRENTLY, TRACKER IS IMPLEMENTED USING INFLUENCE DIAGRAM UTILITIES, AND IS UNDERGOING EVALUATION

K9-7246/062

SUMMARY

This presentation summarized the charcteristics of the Influence Diagram and its applicability to implementing midcourse tracking algorithms. The collection of Influence Diagram utilities perform the probabilistic calculations associated with Kalman filter processing, track spawning, shared contact update and formation update processing, as well as association hypothesis scoring.

Other advantages of the Influence Diagram implementation are the following:

** The Influence Diagram implementation automatically maintains all relevent influences as part of the inferencing process.

** The Kalman Filter implementation is efficient and guarantees a positive semidefinite covariance matrix.

** The Influence Digram provides a useful means to organize and manage the track database.